Algérienne Démocratique et Populaire

Ministère de l’Enseignement Supérieur et de la Recherche Scientifique

Université Constantine 2



Faculté des Nouvelles Technologies de l’Information et la de Communication

Département des Technologies des Logiciels et Systèmes d’Information

Projet de fin d’études pour l’obtention du diplôme de

Licence en Informatique

Option : Génie Logiciel

Thème

Conception et réalisation d’un site web dynamique pour un “Gestion des ventes pour une société immobilière“

Dirigé par : Réalisé par :

-Mohamed Ali Sandeli. - Mouhamed Amine Griche.

- Boudiar Houssam.

- Session Juin 2018 -

1. Introduction :

Dans ce chapitre, nous allons présenter la première étape du processus de développement Pascal Roques. Cette étape consiste à identifier et modéliser les besoins des différents utilisateurs à partir les diagrammes des cas d’utilisations. Les interactions entre les acteurs et le système (au sein des cas d’utilisation) seront explicitées sous forme textuelle (description textuelle), et sous forme graphique au moyen des diagrammes de séquence qui représentent exactement le déroulement de la tâche courante.

1. Etude préliminaire :

L’étude préliminaire est un processus de développement consiste à positionner précisément le champ du système étudié et à faire un premier repérage des besoins dans cette phase nous allons donner une représentation générale de projet :

1. Présentation général du Project :

L’objectif fondamentale de notre site est de réaliser une application web qui consiste à informatiser les différents aspects du fonctionnement d’une gestion des ventes pour une société immobilière et qui permet de :

- Gérer les comptes du clients, agents et operateurs (crée, modifie, supprimer . . .).

- Gérer les rendez-vous (prendre, proposer, annuler . . .).

- Gérer les inscriptions des différents utilisateurs (accepter et refuser les inscriptions).

1. Définition des grands choix techniques :

Pour la réussite de notre projet, nous avons fixé les choix techniques les mieux adaptés :

1. Le processus de développement : Pascal Roques
2. L’outil de modélisation : Le langage UML 2.0.
3. Langages de programmation : J2EE (JSP, SERVLET), Spring Framework, JAVA, JAVA SCRIPT, HTML, CSS, AJAX, JQuery.
4. L’architecture de l’application : basée sur le model Client / Serveur.
5. Le serveur de base de données : MYSQL
6. La plateforme : Windows 8.1 professionnel.
7. Recueil des besoins fonctionnels :

Le site à était implémenté dans le but de réaliser les grandes fonctionnalités. Nous avons classé ces fonctionnalités par type d’utilisateur de la manière suivante :

* L’internaute peut :
* Inscrire.
* Consulter les logements.
* Le Client peut :
* Consulter les logements.
* Consulter les Rendez-vous (Proposer un nouveau RDV, Annuler un RDV ou bien Signaler un RDV).
* L’Agent peut :
* Consulter les logements.
* Noter les opinions des clients.
* L’Opérateur peut :
* Consulter les logements.
* Gérer les comptes des clients.
* L’Administrateur peut :
* Consulter les Statistique.
* Gérer les comptes des Employées.
* Gérer les logements.

1. Définition rapide sur les diagrammes :
2. Le diagramme de cas d’utilisation :

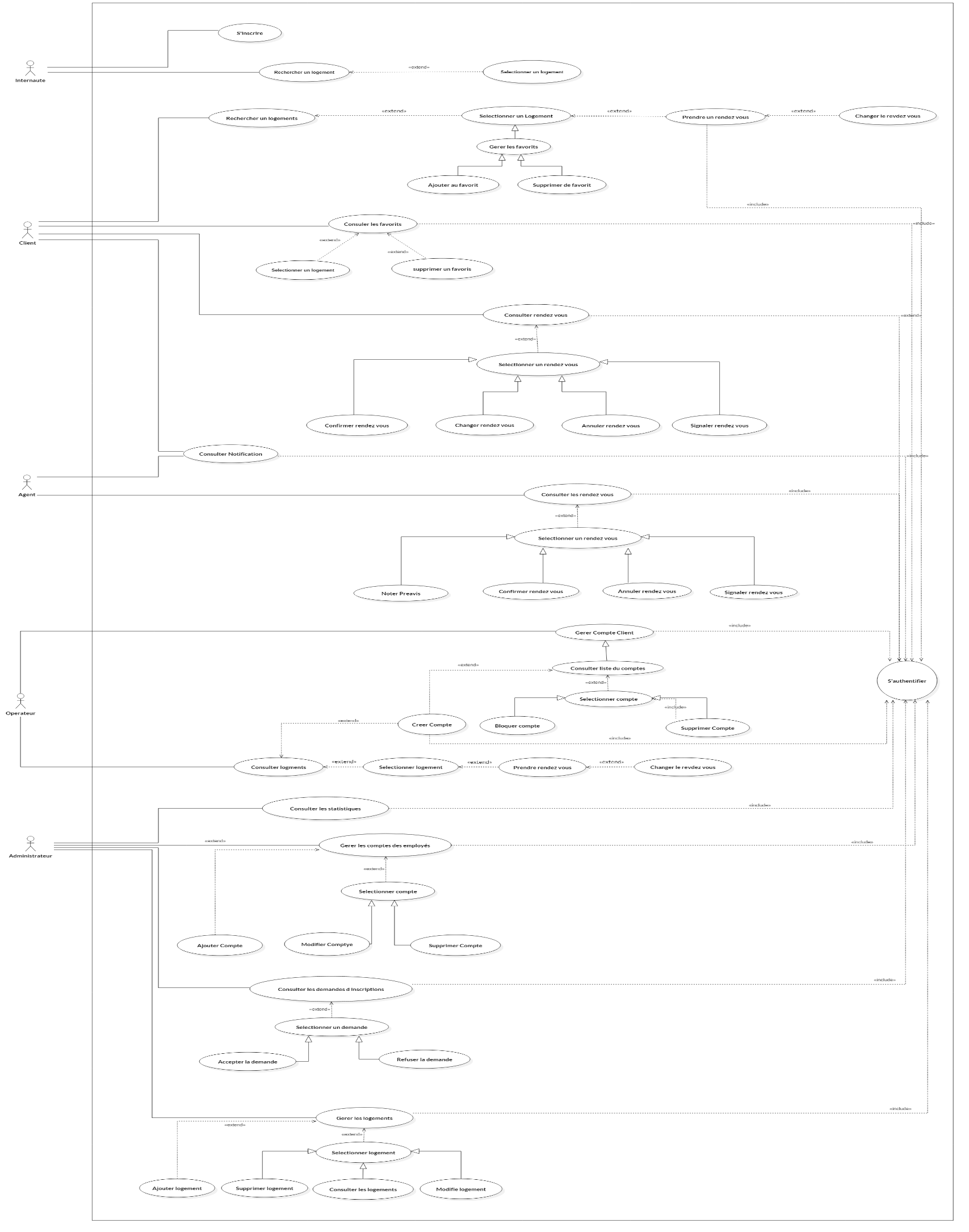
Le diagramme de cas d’utilisation est utilisé dans l’activité de spécification des besoins. Il montre les interactions fonctionnelles entre les acteurs et le système à l’étude.

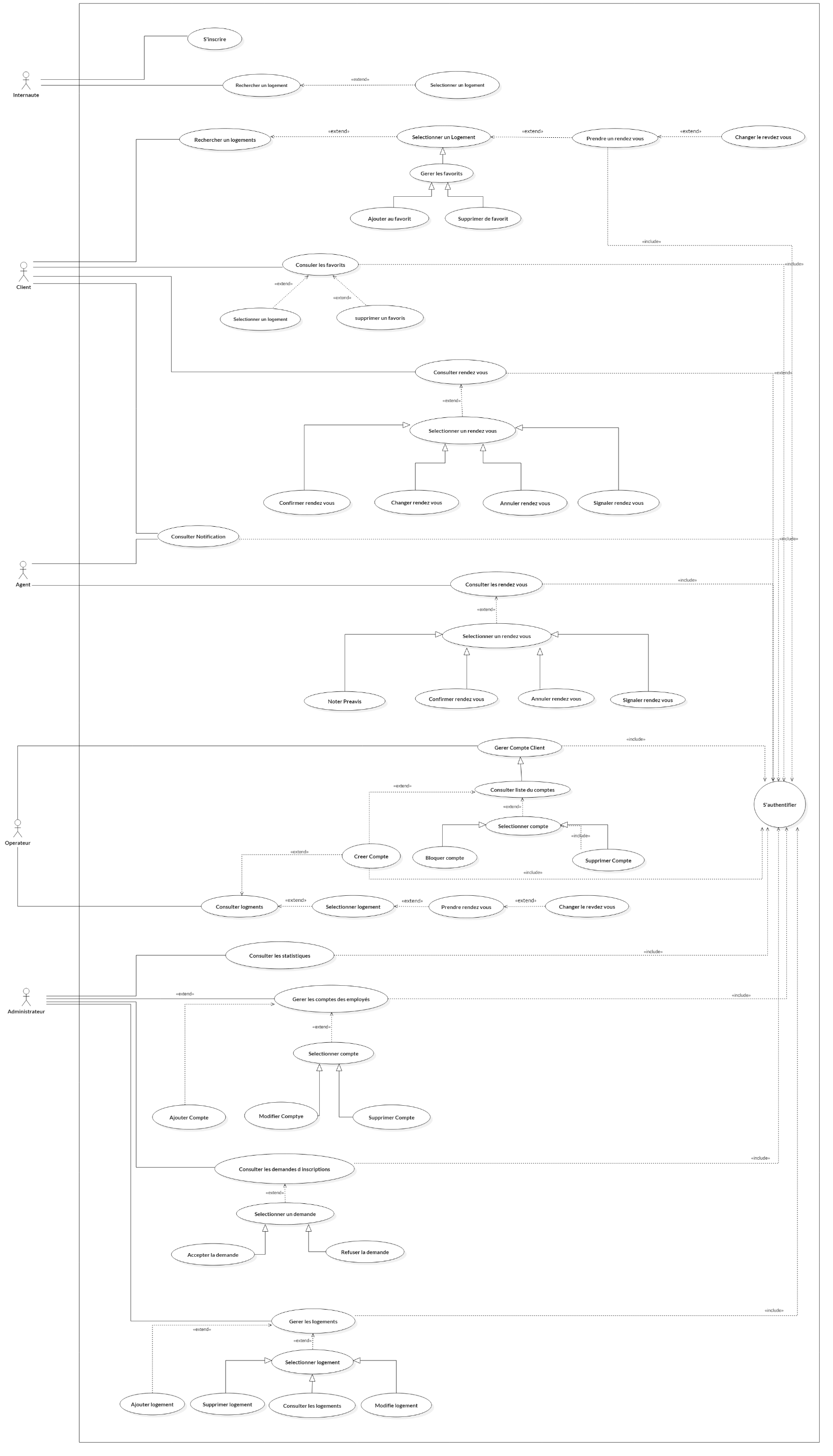
1. Les diagrammes de séquence système :

Les diagrammes de séquences mettent en valeur les échanges de messages (déclenchant des événements) entre acteurs et objets (ou entre objets et objets) de manière chronologique, l’évolution du temps se lisant de haut en bas. Chaque colonne correspond à un objet (décrit dans le diagramme des classes), ou éventuellement à un acteur, introduit dans le diagramme des cas. La ligne de vie de l’objet représente la durée de son interaction avec les autres objets du diagramme.

4- Diagramme de cas d’utilisation :

Représente un ensemble de séquence d'action qui sont réalisées par le système et qui produisent un résultat observable intéressant pour un acteur particulier.





1. Les Fiches Descriptives et leurs diagrammes de séquence :

**Cas d’utilisation : Inscrire**

- Fiche descriptive :

|  |  |
| --- | --- |
| Cas d’utilisation | Inscrire |
| Acteur | Internaute |
| Type | Primaire |
| Objectif | Permet au visiteur de s’inscrire dans le site |
| Préconditions | Néant |
| Post conditions | L’internaute dispose un compte |
| Scénario Nominal | 1. L’utilisateur demande le formulaire d’inscription. 2. Le système affiche le formulaire d’inscription. 3. L’internaute remplit le formulaire. 4. L’internaute envoie le formulaire. 5. Le système enregistre le nouveau compte. 6. Le système affiche la page d’accueil du nouveau l’utilisateur. |
| Scénario Alternatif | 3.a Information invalide.  a.1. Le système affiche un message d’erreur.  Reprendre à l’étape 2 |
| Scénario Exception | - Annuler l’envoie.  - Compte existe déjà. |

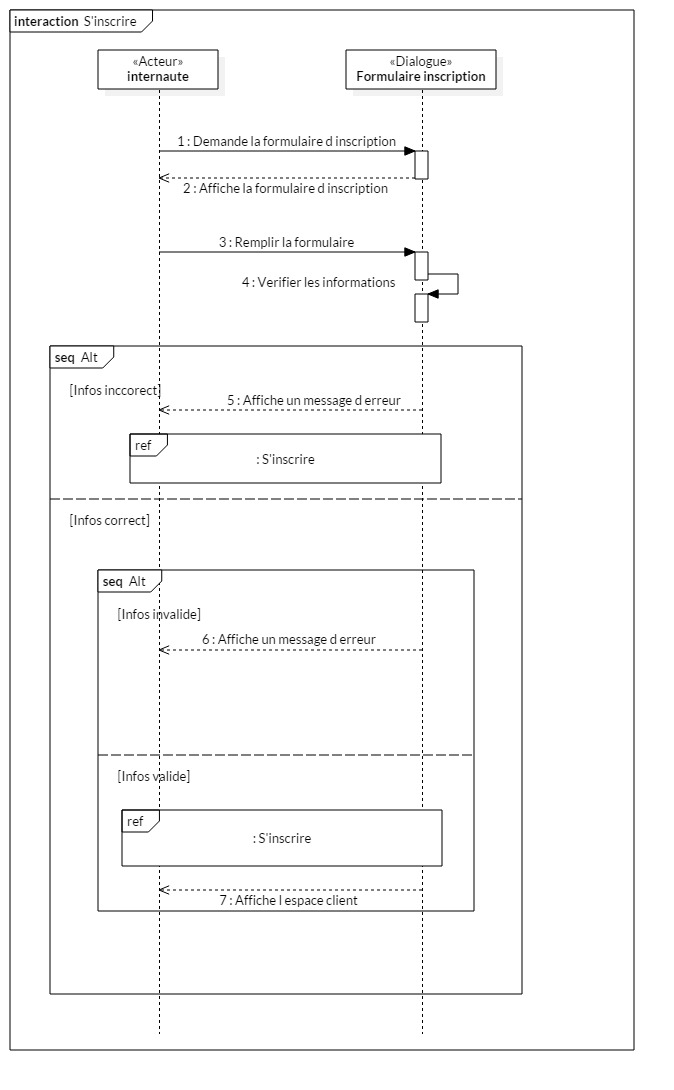
 - Diagramme de séquence :

Figure 2

- Cas d’utilisation : Authentifier

|  |  |
| --- | --- |
| Cas d’utilisation | S’authentifier |
| Acteur | Client , Operateur , Agent , Administrateur |
| Type | Secondaire |
| Objectif | Accéder à son propre espace. |
| Préconditions | Avoir un pseudo (email ou nom d’utilisateur)et un mot de passe . |
| Post conditions | Le client accédé a son espace de travail. |
| Scénario Nominal | 1. L’utilisateur demande le formulaire d’authentification. 2. Le système affiche le formulaire d’authentification. 3. L’utilisateur saisit son pseudo et mot de passe. 4. Le système vérifie son existence. 5. Le système affiche à l’utilisateur son espace de travail. |
| Scénario Alternatif |  |
| Scénario Exception | - Pseudo ou mot de passe invalide.  - Annulation. |

- Fiche descriptive :

- Diagramme de séquence :

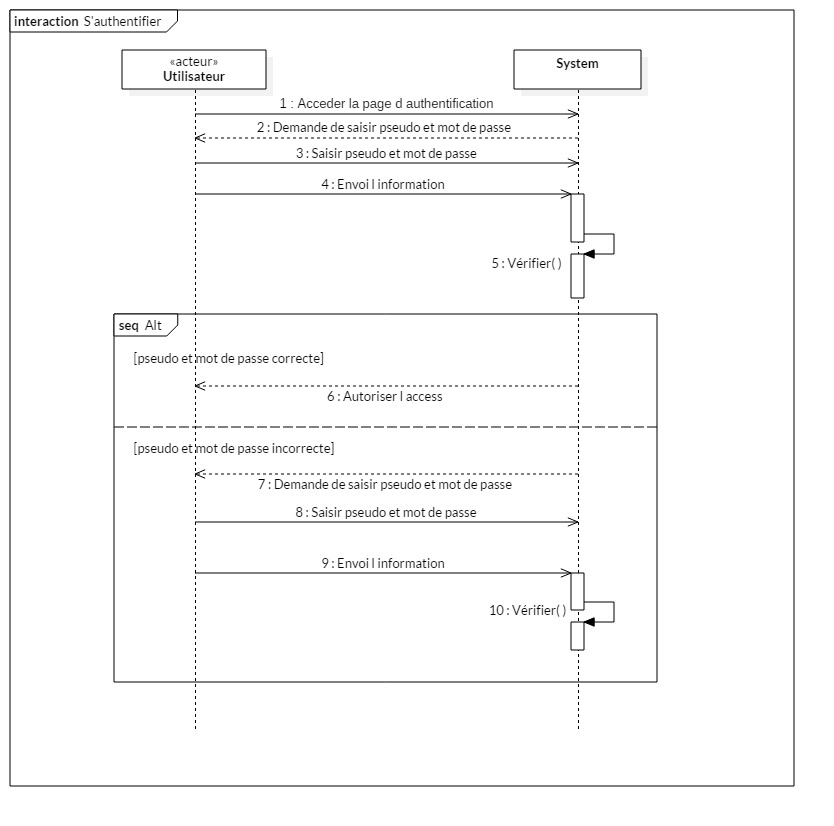


Figure 3

- Cas d’utilisation : Consulter les logements

- Fiche descriptive :

|  |  |
| --- | --- |
| Cas d’utilisation | Rechercher un logement |
| Acteur | Internaute , Client , Operateur |
| Type | Primaire |
| Objectif | Permet au Client de voir les logements disponible |
| Préconditions | Néant |
| Post conditions | Afficher les logements disponible |
| Scénario Nominal | 1. L’utilisateur accéder l’espace de recherche. 2. Le système affiche l’espace de recherche. 3. L’utilisateur remplit les critères de recherche pour filtrer les logements. 4. Le système renvoie la liste des logements qui convient les critères. |
| Scénario Alternatif | 4.a Aucun logement était trouvé.  a.1. Le système affiche une liste vide.  Reprendre à l’étape 2 |
| Scénario Exception | - L’utilisateur Annule la demande. |

- Diagramme de séquence :

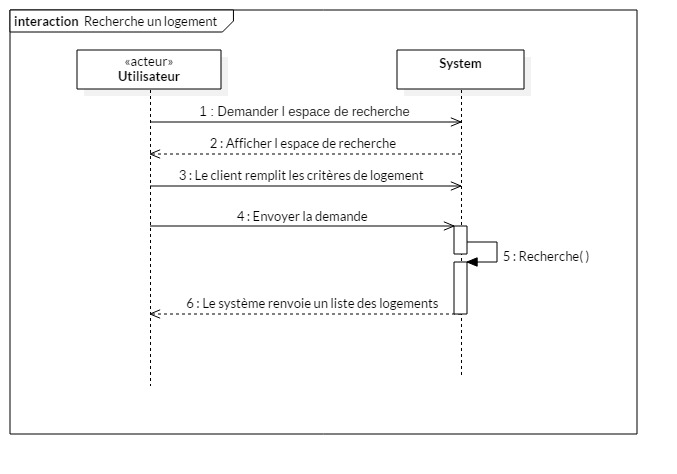


Figure 4

- Cas d’utilisation : Sélectionner un logement

- Fiche descriptive :

|  |  |
| --- | --- |
| Cas d’utilisation | Sélectionner un logement |
| Acteur | Internaute , Client , Operateur |
| Type | Primaire |
| Objectif | Permet au Client de voir les détails d’un logements |
| Préconditions | Consulter les logements |
| Post conditions | Afficher les détails du logements choisi. |
| Scénario Nominal | 1. L’utilisateur demande la liste des logements. 2. Le système affiche la liste des logements. 3. Le client choisi un logement du la liste. 4. Le système affiche les détailles du logement choisi. |
| Scénario Alternatif |  |
| Scénario Exception | - L’utilisateur Annule la demande. |

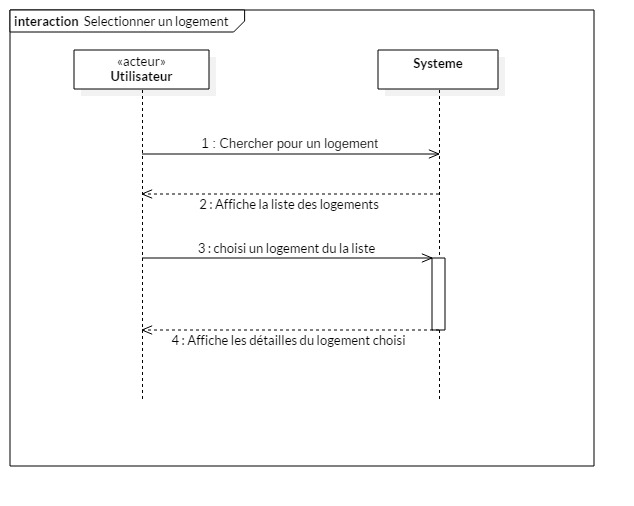
 - Diagramme de séquence :

Figure 5

- Cas d’utilisation : Prendre un rendez-vous

- Fiche descriptive :

|  |  |
| --- | --- |
| Cas d’utilisation | Prendre un RDV |
| Acteur | Client , Operateur |
| Type | Primaire |
| Objectif | Permet au Client de prendre un rendez-vous. |
| Préconditions | - Sélectionner un logement.  - S’authentifier. |
| Post conditions | Afficher les détails d’un rendez-vous. |
| Scénario Nominal | 1. Le client demande un rendez-vous. 2. Le système lui génère un rendez-vous. 3. Le client accepter le rendez-vous. |
| Scénario Alternatif |  |
| Scénario Exception | - L’utilisateur Annule l’opération. |

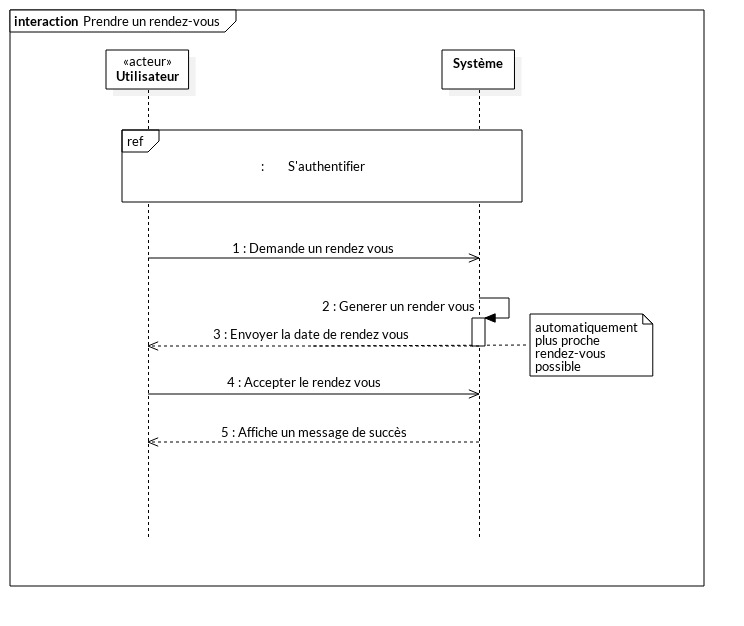
 - Diagramme de séquence :

Figure 6

- Cas d’utilisation : Confirmer un rendez-vous

- Fiche descriptive :

|  |  |
| --- | --- |
| Cas d’utilisation | Confirmer rendez-vous |
| Acteur | Client , Agent |
| Type | Primaire |
| Objectif | Permet au utilisateur de valider son présence . |
| Préconditions | S’authentifier |
| Post conditions | Le rendez-vous a été confirmer par l’utilisateur |
| Scénario Nominal | 1. Le système demande aux utilisateur de confirmer le rendez-vous. 2. L’utilisateur confirmer le rendez-vous. 3. Le système enregistre l’opération. 4. Le système notifie les utilisateurs concernant. |
| Scénario Alternatif |  |
| Scénario Exception | - Annulation. |

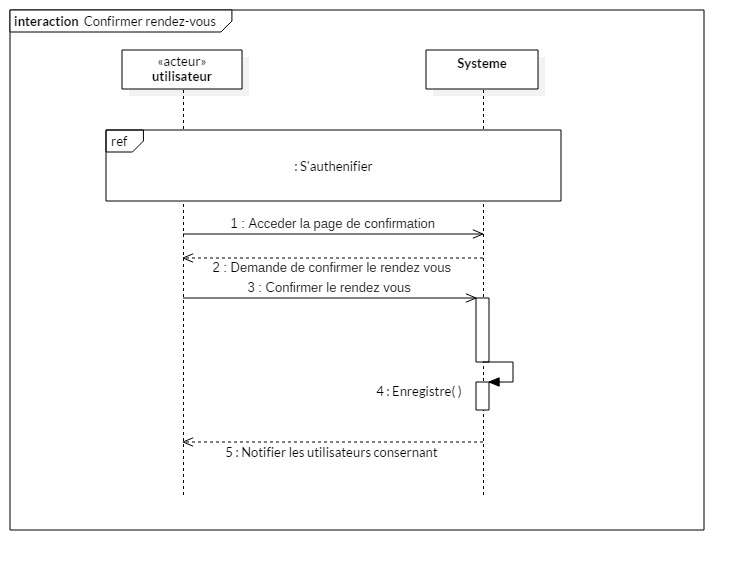
 - Diagramme de séquence :

Figure 7

- Cas d’utilisation : Noter les préavis

- Fiche descriptive :

|  |  |
| --- | --- |
| Cas d’utilisation | Noter les préavis. |
| Acteur | Agent. |
| Type | Secondaire. |
| Objectif | Prendre l’avis du client. |
| Précondition | S’authentifier. |
| Post-Condition | Obtient l’avis finale d’un client. |
| Scenario nominal | 1. L’agent consulte un RDV. 2. L’agent demande de noter un préavis. 3. Le système affiche un formulaire du préavis. 4. L’agent remplir le formulaire. 5. L’agent envoi le formulaire au système. 6. Le système enregistre la préavis. |
| Scenario Alternatif | 2.a.1 Les informations invalide  a.2 Le system renvoi un message de l’erreur  Reprendre a l’étape 2. |
| Scenario Exception | L’agent annulé l’opération. |

- Diagramme de séquence :

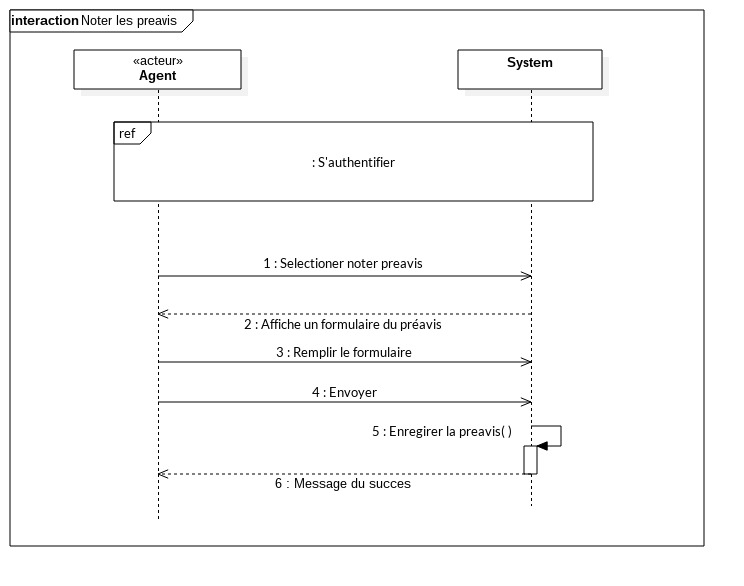


Figure 8

- Cas d’utilisation : Signaler un rendez vous

- Fiche descriptive :

|  |  |
| --- | --- |
| Cas d’utilisation | Signaler le RDV |
| Acteur | Client , Opérateur , Agent |
| Type | Primaire |
| Objectif | Permet au utilisateur de signaler le RDV sélectionnée. |
| Préconditions | - S’authentifier. |
| Post conditions | Envoyer un formulaire de la proclamation d’un RDV. |
| Scénario Nominal | 1. L’Utilisateur consulter un RDV. 2. L’Utilisateur demande de signaler le RDV. 3. Le système enregistre le signal. 4. Le système informe l’opérateur au signal. |
| Scénario Alternatif |  |
| Scénario Exception | - L’utilisateur Annule la demande. |

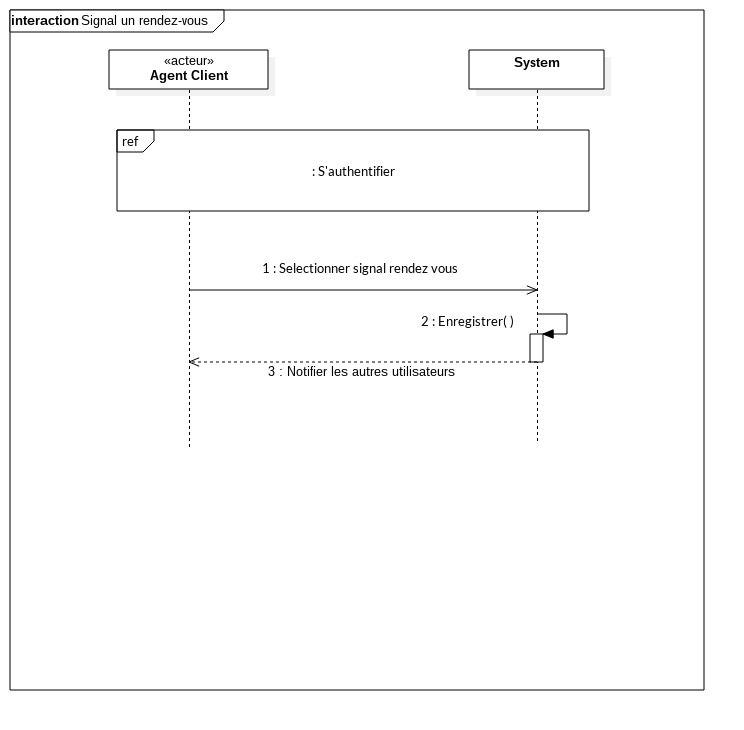
 - Diagramme de séquence :

Figure 9

- Cas d’utilisation :  Annuler un rendez-vous

- Fiche descriptive :

|  |  |
| --- | --- |
| Cas d’utilisation | Annuler le RDV |
| Acteur | Client , Opérateur |
| Type | Primaire |
| Objectif | Permet au utilisateur d’annuler le RDV sélectionnée. |
| Préconditions | - S’authentifier. |
| Post conditions | L’annulation d’un RDV. |
| Scénario Nominal | 1. L’Utilisateur demande d’annuler le rendez-vous. 2. Le système demande de confirme l’annulation. 3. L’utilisateur confirme l’annulation du rendez-vous. 4. Le système valide l’annulation. 5. Le système notifier les autres utilisateurs concertants. |
| Scénario Alternatif |  |
| Scénario Exception | - L’utilisateur Annule la demande. |

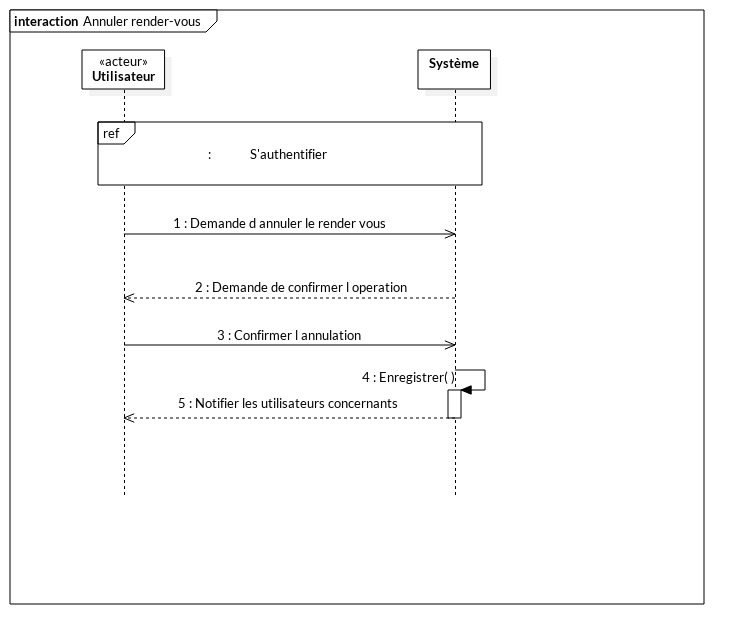
 - Diagramme de séquence :

Figure 10

- Cas d’utilisation :  Créer un compte client

- Fiche descriptive :

|  |  |
| --- | --- |
| Cas d’utilisation | Créer un compte |
| Acteur | Operateur |
| Type | Primaire |
| Objectif | Permet à l’opérateur d’inscrit un nouveau client |
| Préconditions | -S’authentifier.  -L’operateur reçoit l’appelle du client.  -Gere les comptes. |
| Post conditions | Avait un nouveau compte après la demande d’un client. |
| Scénario Nominal | 1. L’opérateur demande de créer un compte. 2. Le système affiche un formulaire de la création. 3. L’opérateur saisit les informations du compte. 4. L’opérateur envoyer le formulaire. 5. Le système enregistre le nouveau compte. |
| Scénario Alternatif | 3.a Information invalide.  a.1. Le système affiche un message d’erreur.  Reprendre à l’étape 2 |
| Scénario Exception | - Annuler l’envoie.  - Compte existe déjà. |

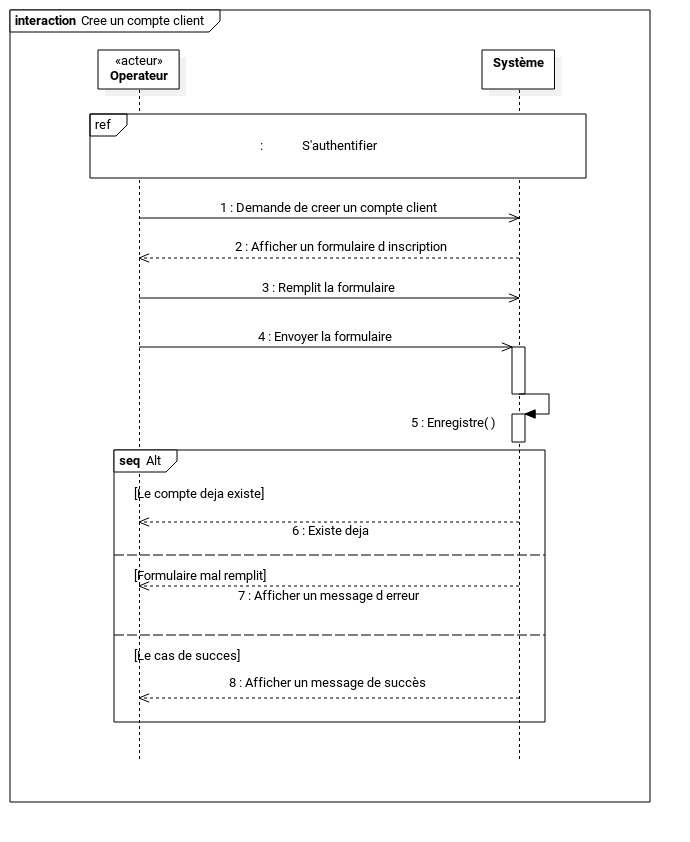
 - Diagramme de séquence :

Figure 11

- Cas d’utilisation :  Accepter l’inscription

- Fiche descriptive :

|  |  |
| --- | --- |
| Cas d’utilisation | Accepter les inscriptions |
| Acteur | Administrateur |
| Type | Secondaire |
| Objectif | Valider l’inscriptions des operateur et agents |
| Précondition | - S’authentifier  - Gérer les comptes des employés |
| Post-Condition | Vérifier tous les nouveau comptes des employées dans le système. |
| Scenario nominal | 1. L’utilisateur demande de valider les inscriptions.  2.Le system affiche la liste des comptes qui non pas encore valide.  2. L’admin sélectionne un compte de la liste.  3. Le système affiche les détails de compte.  4. L’admin valide l’inscription de cet utilisateur.  5. Le système affiche un message de succès. |
| Scenario Alternatif |  |
| Scenario Exception | - Annuler l’envoie. |

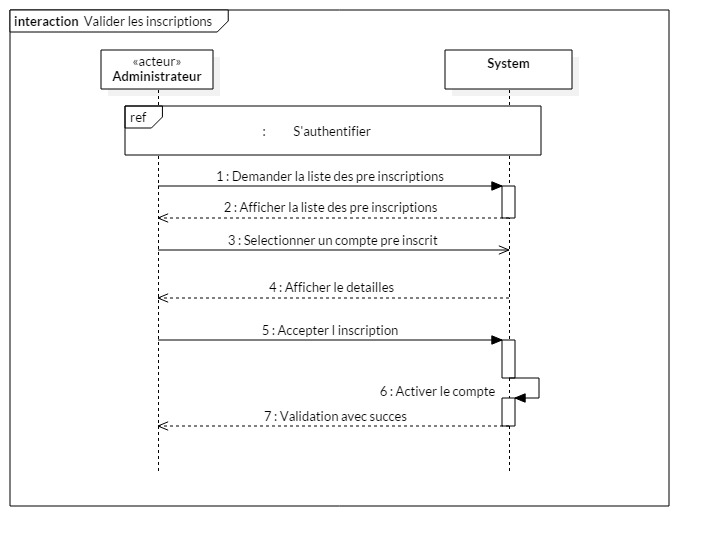
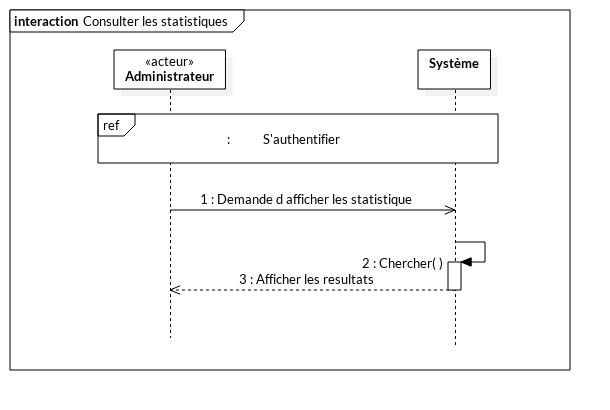
 - Diagramme de séquence :

Figure 12

- Cas d’utilisation :  Consulter les statistiques

- Fiche descriptive :

|  |  |
| --- | --- |
| Cas d’utilisation | Consulter les statistique |
| Acteur | Administrateur |
| Type | Secondaire |
| Objectif | Permet à l’admin de voir tous les statistique des logements et employées |
| Préconditions | S’authentifie |
| Post conditions | Afficher les statistique de l’agence. |
| Scénario Nominal | 1. Le système demande une date de début et fin. 2. L’administrateur remplit les dates. 3. Le système renvoie la liste des statistique dans cette période. |
| Scénario Alternatif | 4.a Date invalide.  a.1. Le système affiche un message d’erreur.  Reprendre à l’étape 2 |
| Scénario Exception | - L’utilisateur Annule la demande. |

 - Diagramme de séquence :

\*

Figure 13

- Cas d’utilisation :  Ajouter un logement

- Fiche descriptive :

|  |  |
| --- | --- |
| Cas d’utilisation | Ajouter un logement |
| Acteur | Administrateur BD |
| Type | Secondaire |
| Objectif | Ajouter un logements au base de données. |
| Précondition | - S’authentifier |
| Post-Condition | un nouveau logements est ajoutée dans le base de données. |
| Scenario nominal | 1. Le system affiche une formulaire d’ajout.  2. L’admin remplit le formulaire est envoyer là.  3. Le système enregistre le logement.  5. Le système affiche un message de succès. |
| Scenario Alternatif |  |
| Scenario Exception | - Annuler l’envoie. |

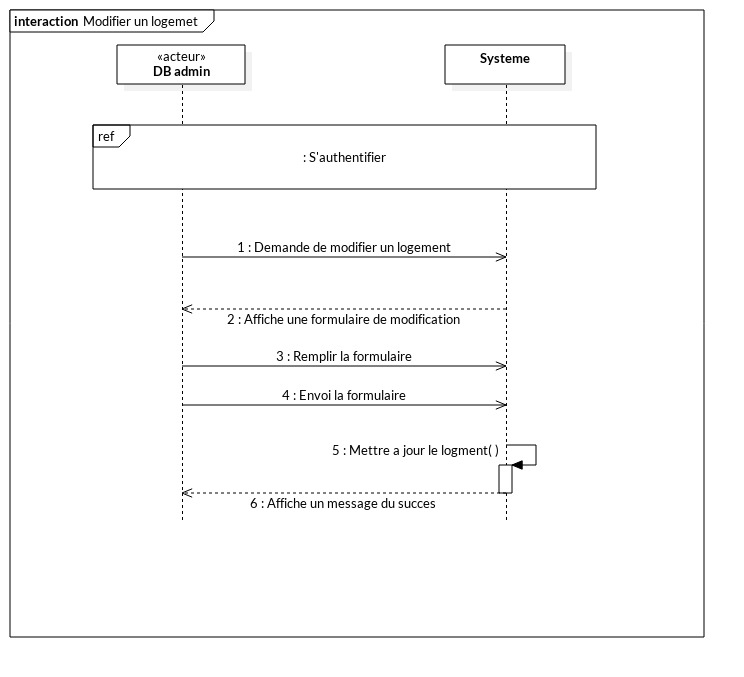
 - Diagramme de séquence :

Figure 14

- Cas d’utilisation :  Modifier un logement

- Fiche descriptive :

|  |  |
| --- | --- |
| Cas d’utilisation | Modifier un logement |
| Acteur | Administrateur BD |
| Type | Secondaire |
| Objectif | Permet de modifier des critères d’un logements dans base de données. |
| Précondition | - S’authentifier |
| Post-Condition | un logement a est modifier dans le base de données. |
| Scenario nominal | 1. Le system affiche un formulaire de modification.  2. L’admin remplit le formulaire est envoyer là.  3. Le système enregistre le logement.  5. Le système affiche un message de succès. |
| Scenario Alternatif |  |
| Scenario Exception | - Annuler l’envoie. |

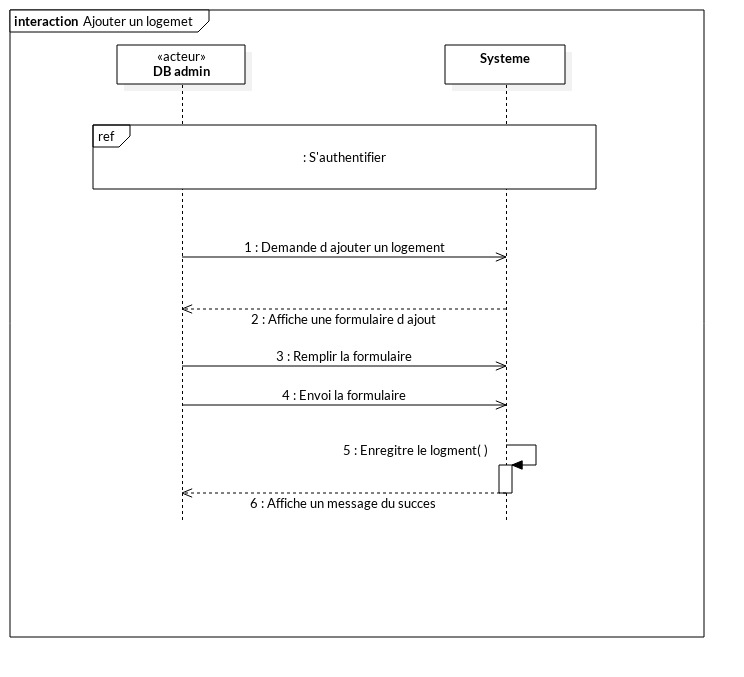
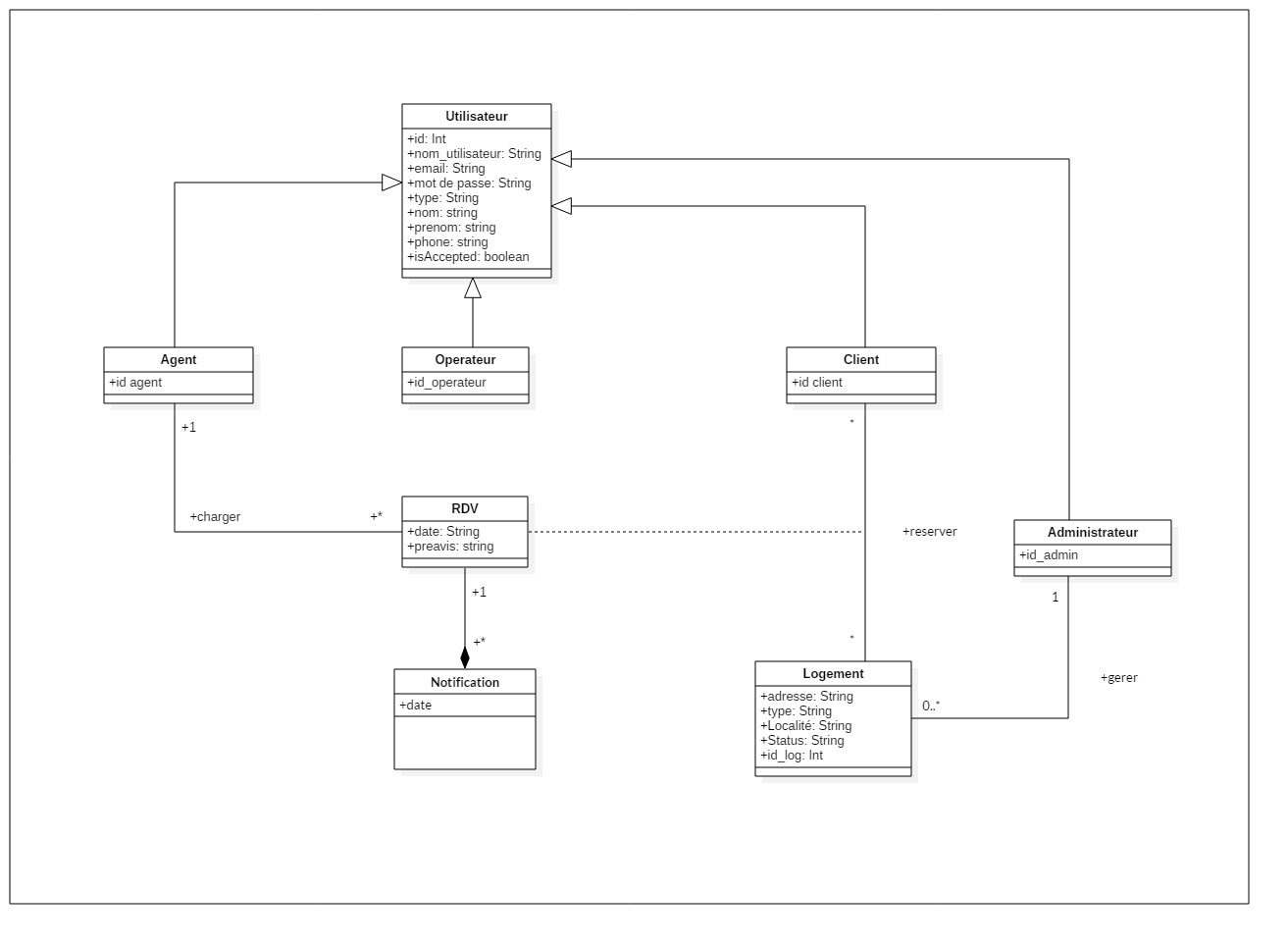
 - Diagramme de séquence :

Figure 15

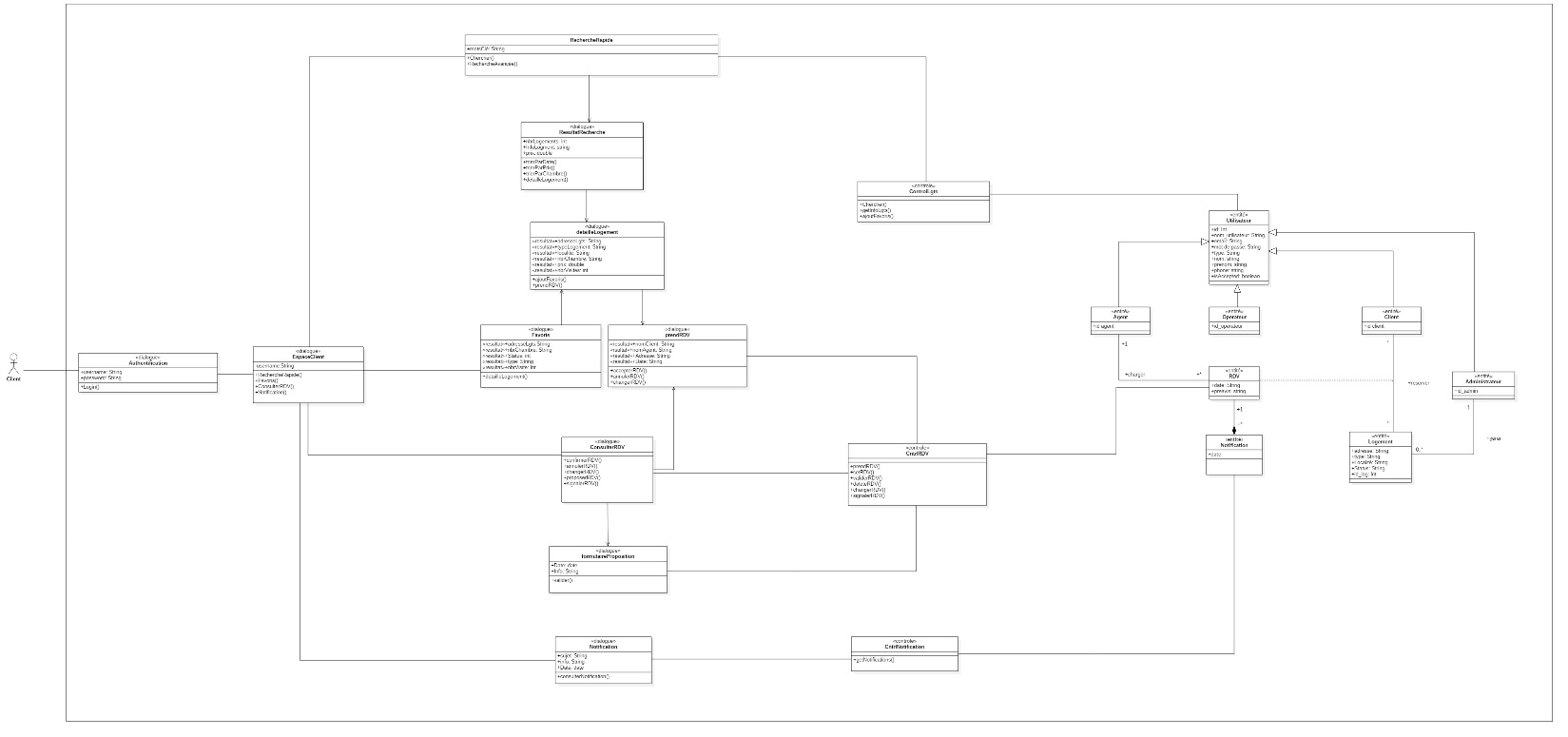
7- Conclusion

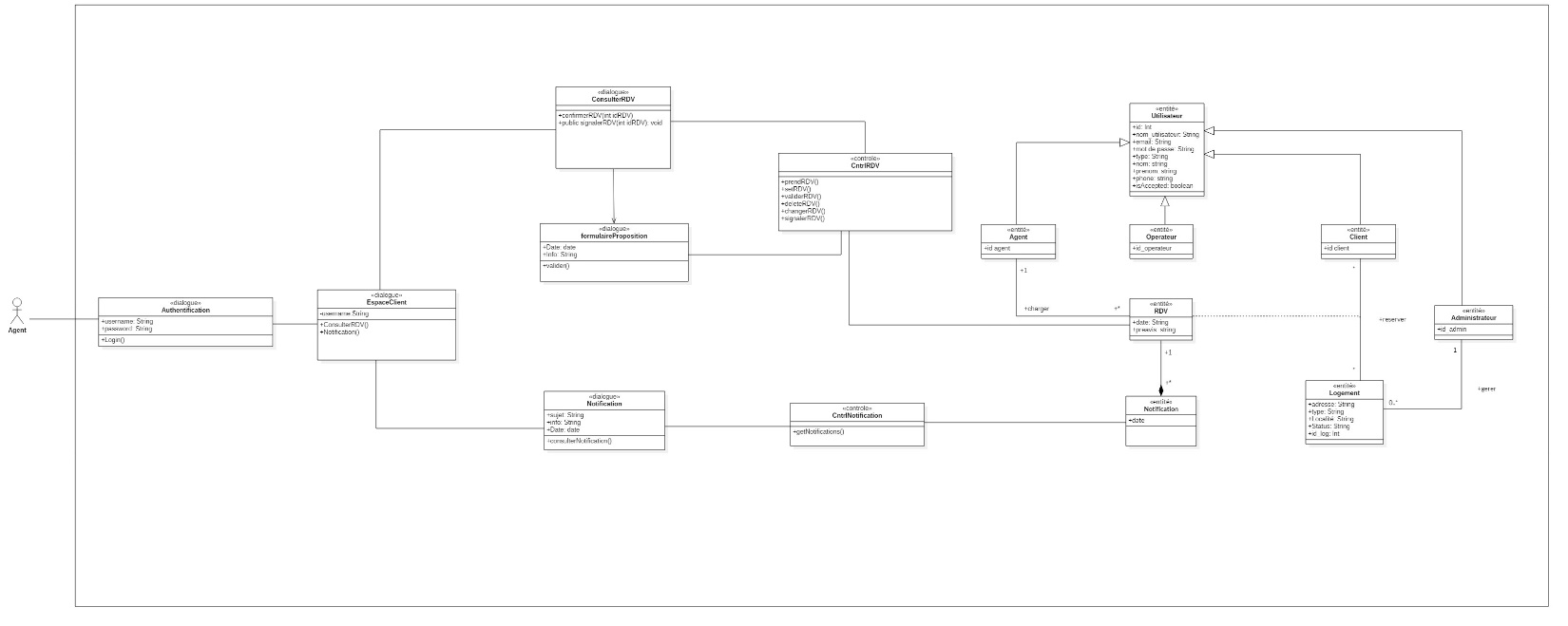
Ce chapitre a été consacré à la présentation de notre conception en appliquant la démarche UP et les fiches descriptives avec les différents diagrammes UML (DCU, DSE) ces derniers représentent une vue nécessaire pour notre projet, de plus les représentations graphiques nous permettront d’entamer l’implémentation de notre système, que nous allons voir dans le chapitre suivant.

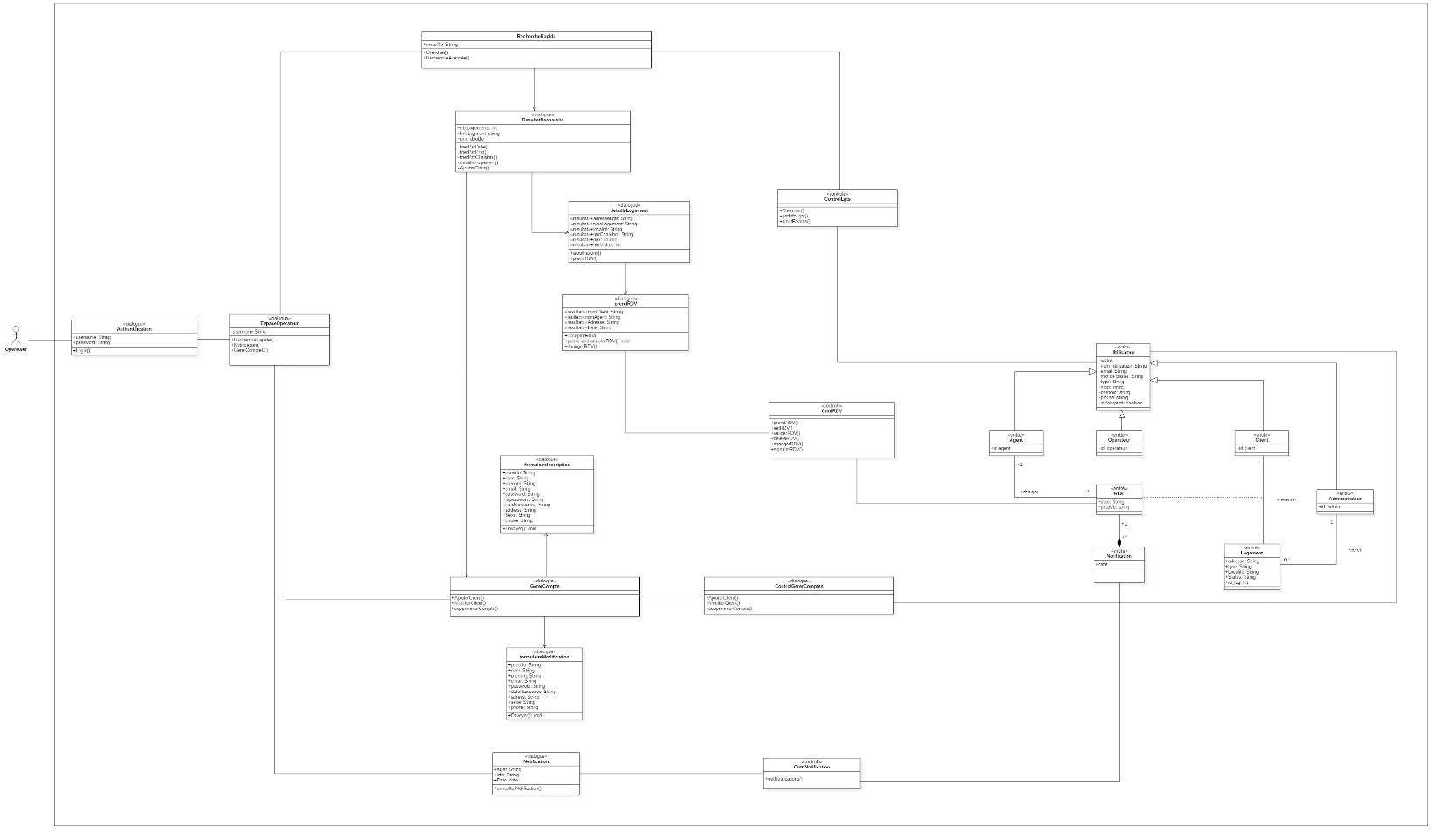
1. Modelé de domaine :

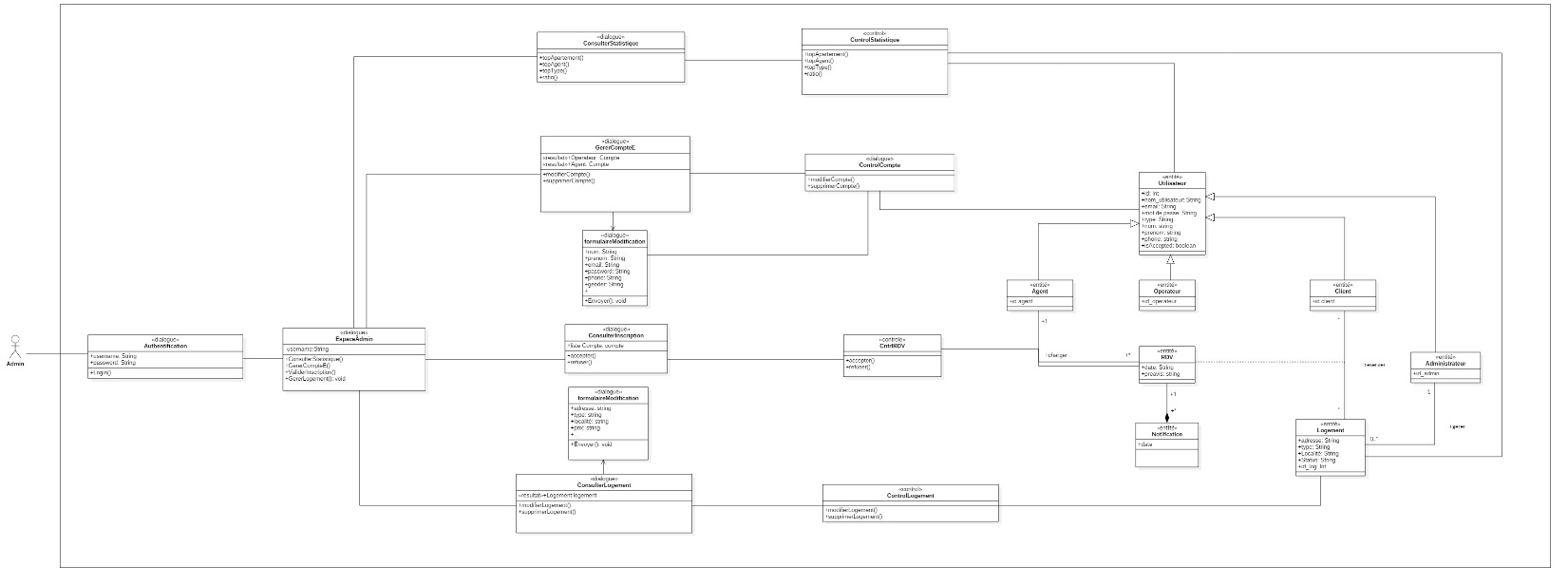
 C'est une représentation visuelle des classes conceptuelles ou des objets du monde réel dans un **domaine** donné. Il est aussi nommé **modèle** conceptuel, **modèle** objet du **domaine**, **modèle** objet d'analyse ou encore diagramme de classes conceptuelles.

2-Les Diagrammes des classes participantes :

2-1. Client :

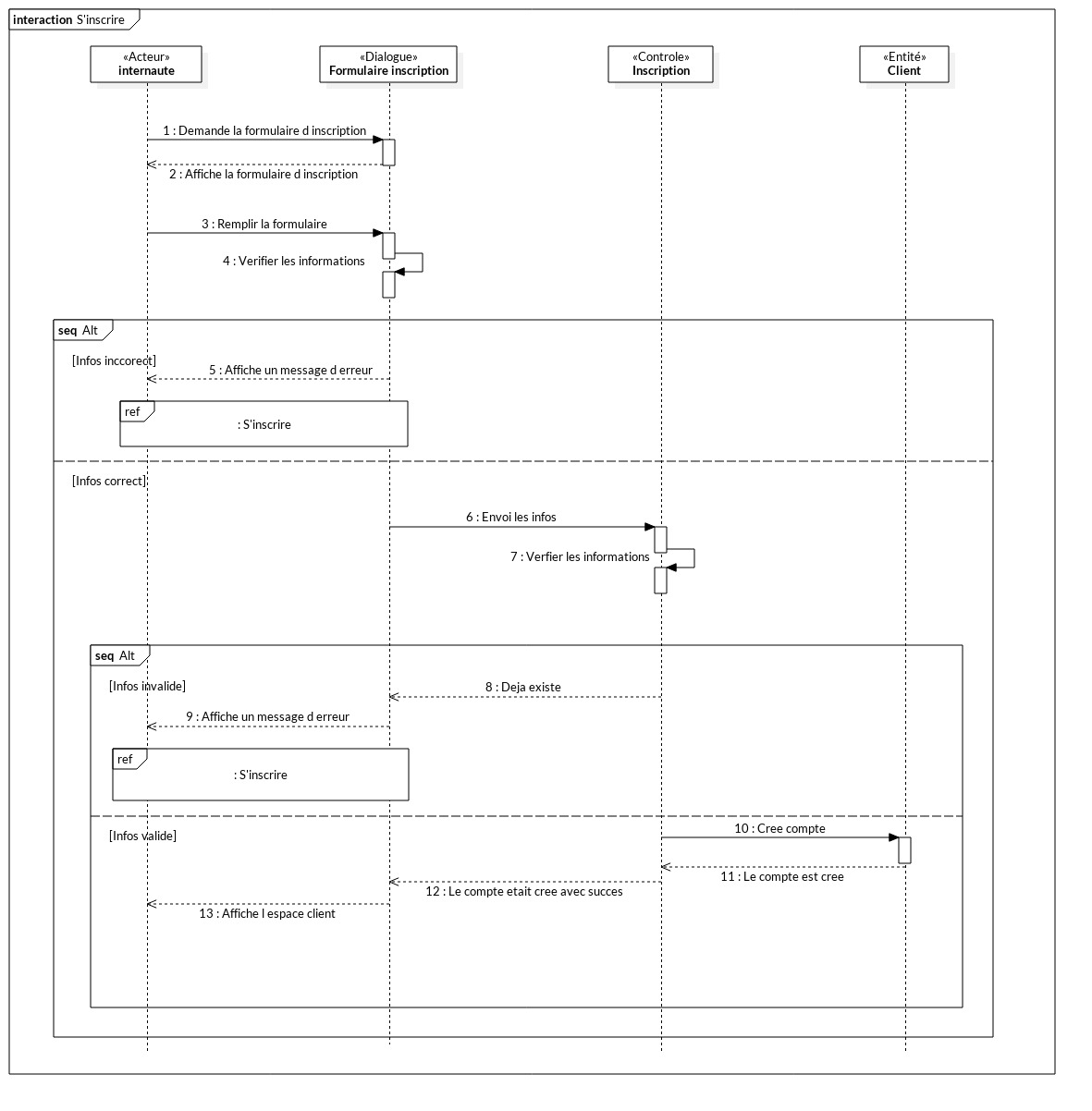
2.2- Agent :

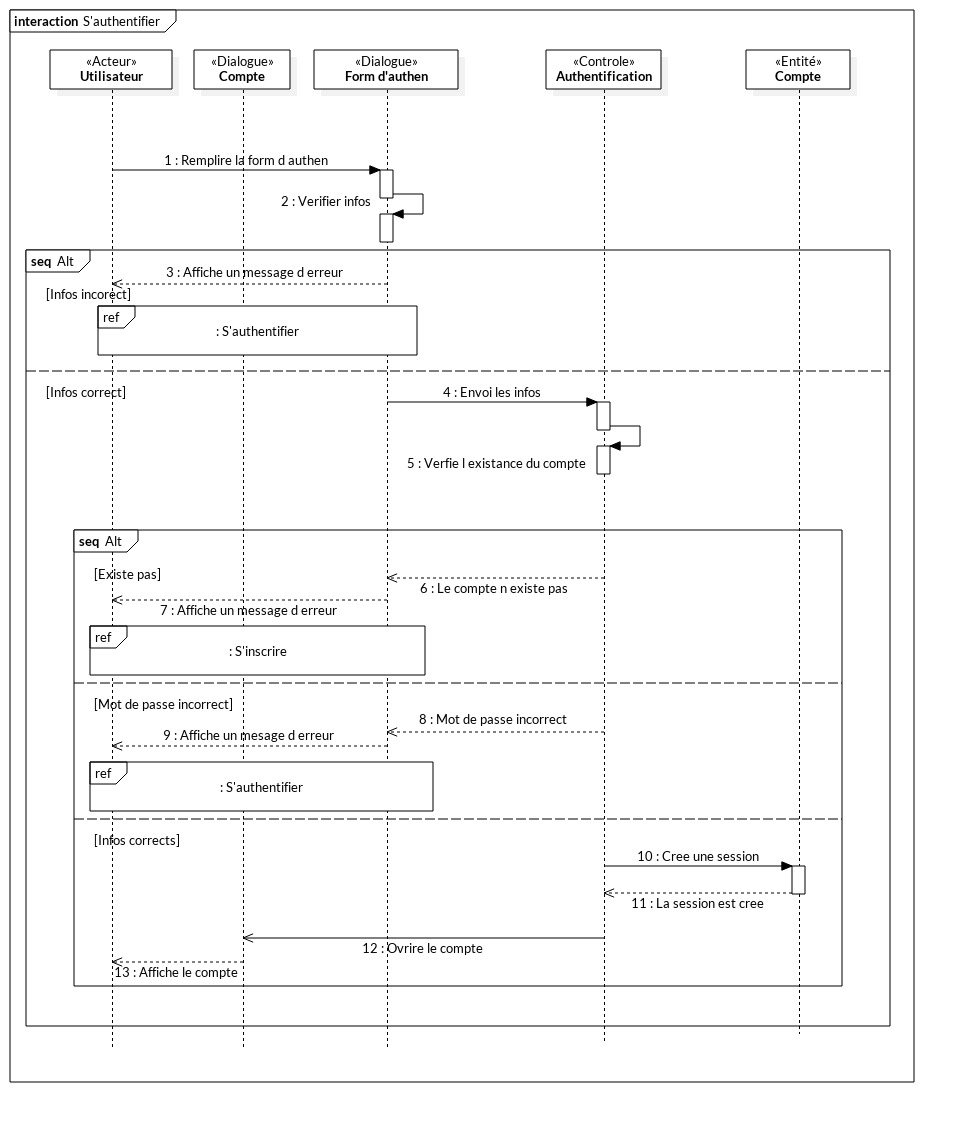
2.3- Operateur :

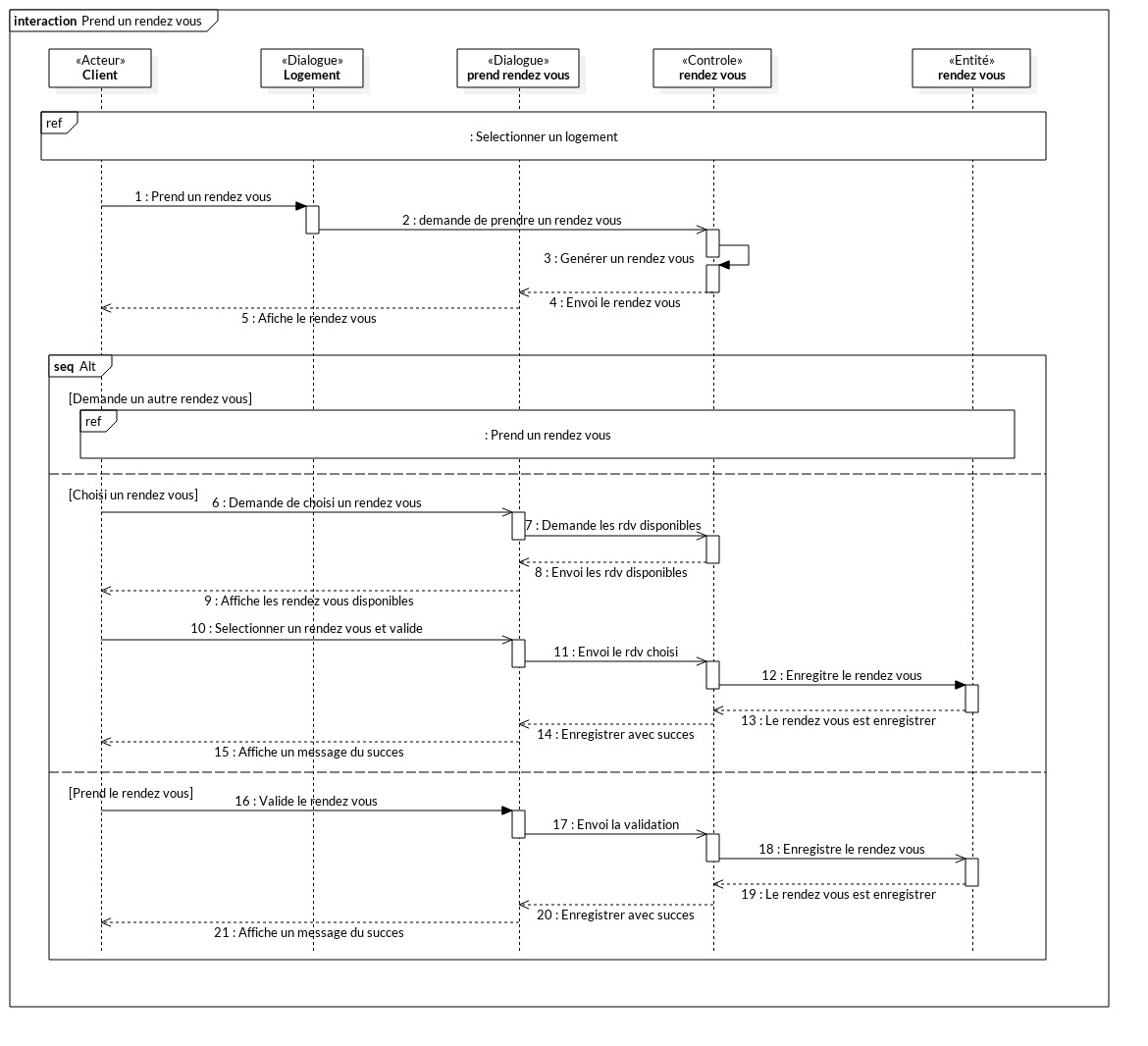
2.4-Administrateur :

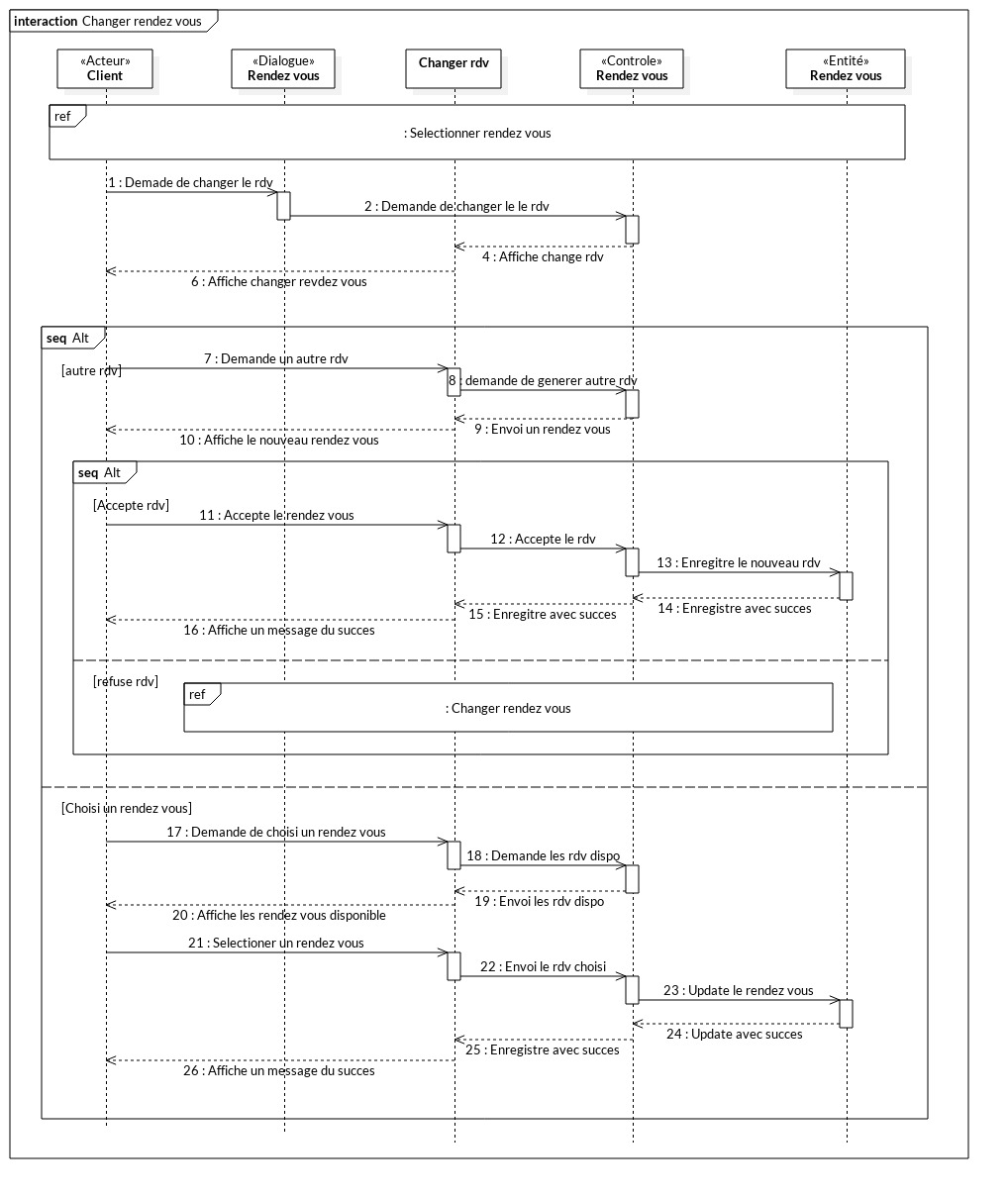
3- Les Diagrammes de séquence :

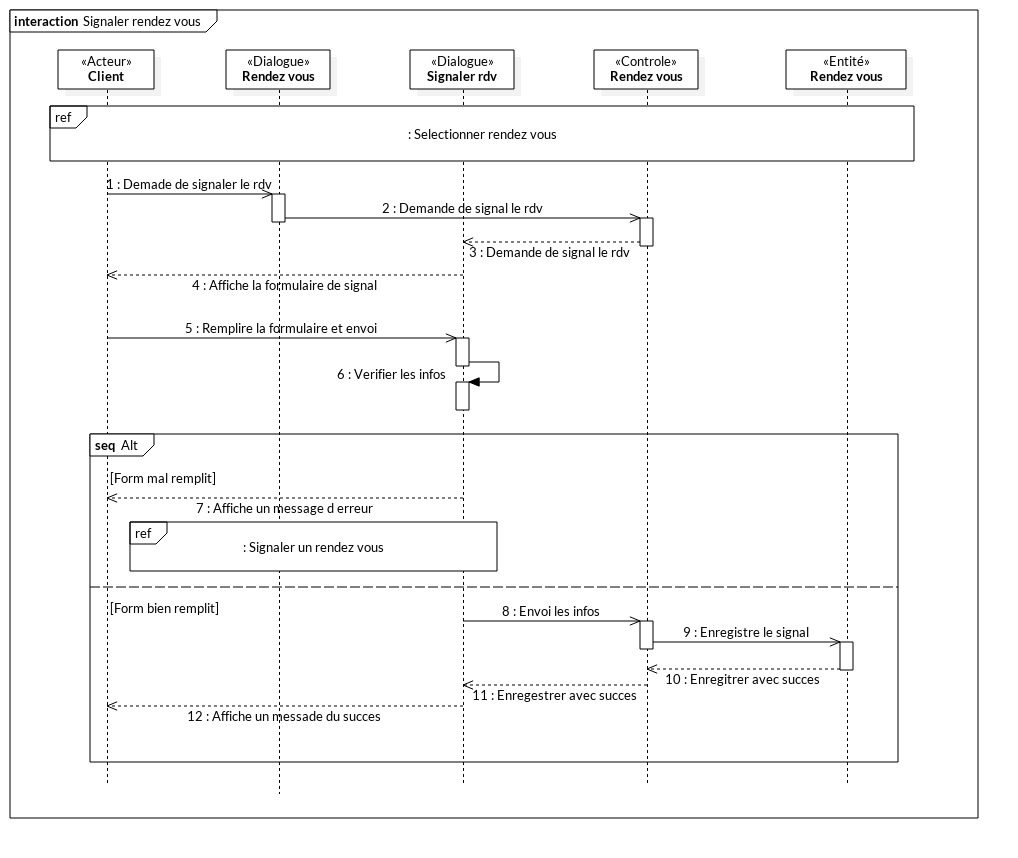
3.1- Inscrire :

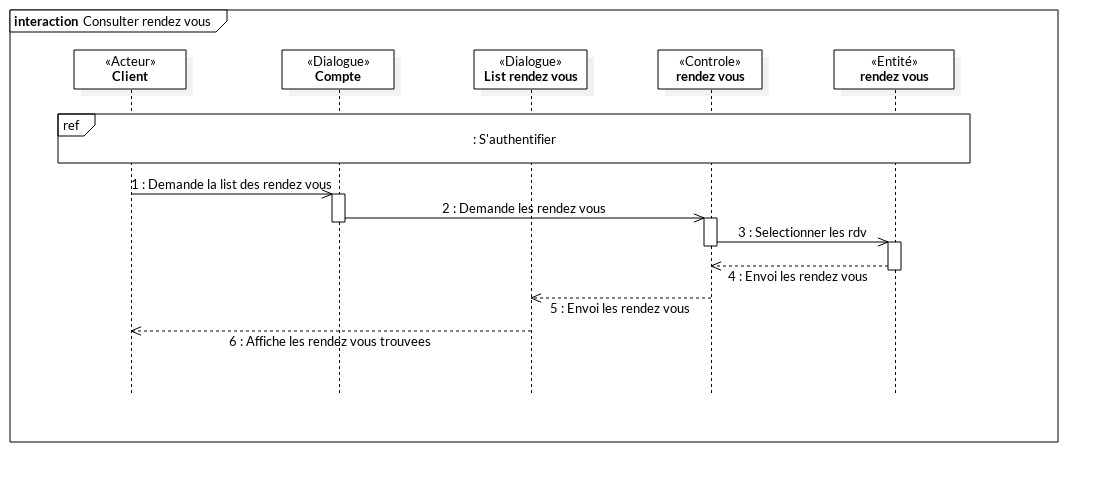


3.2-Authentification :

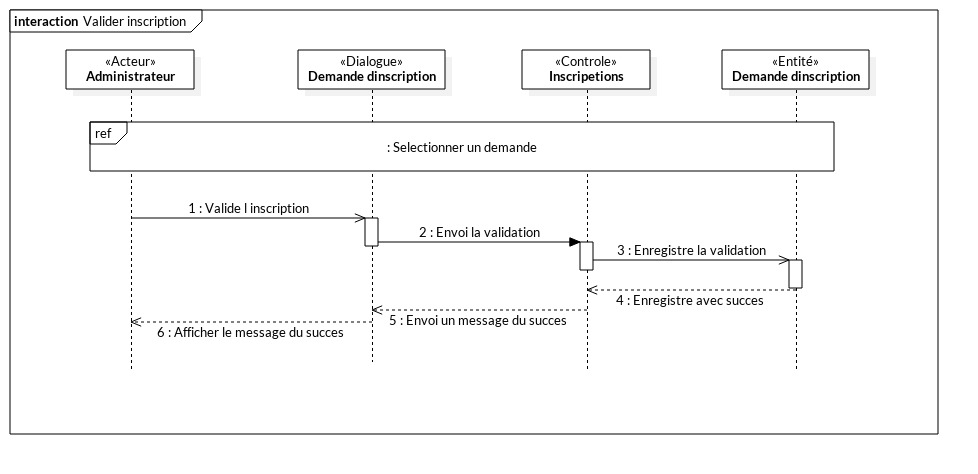
3.3- Prend rendez-vous :

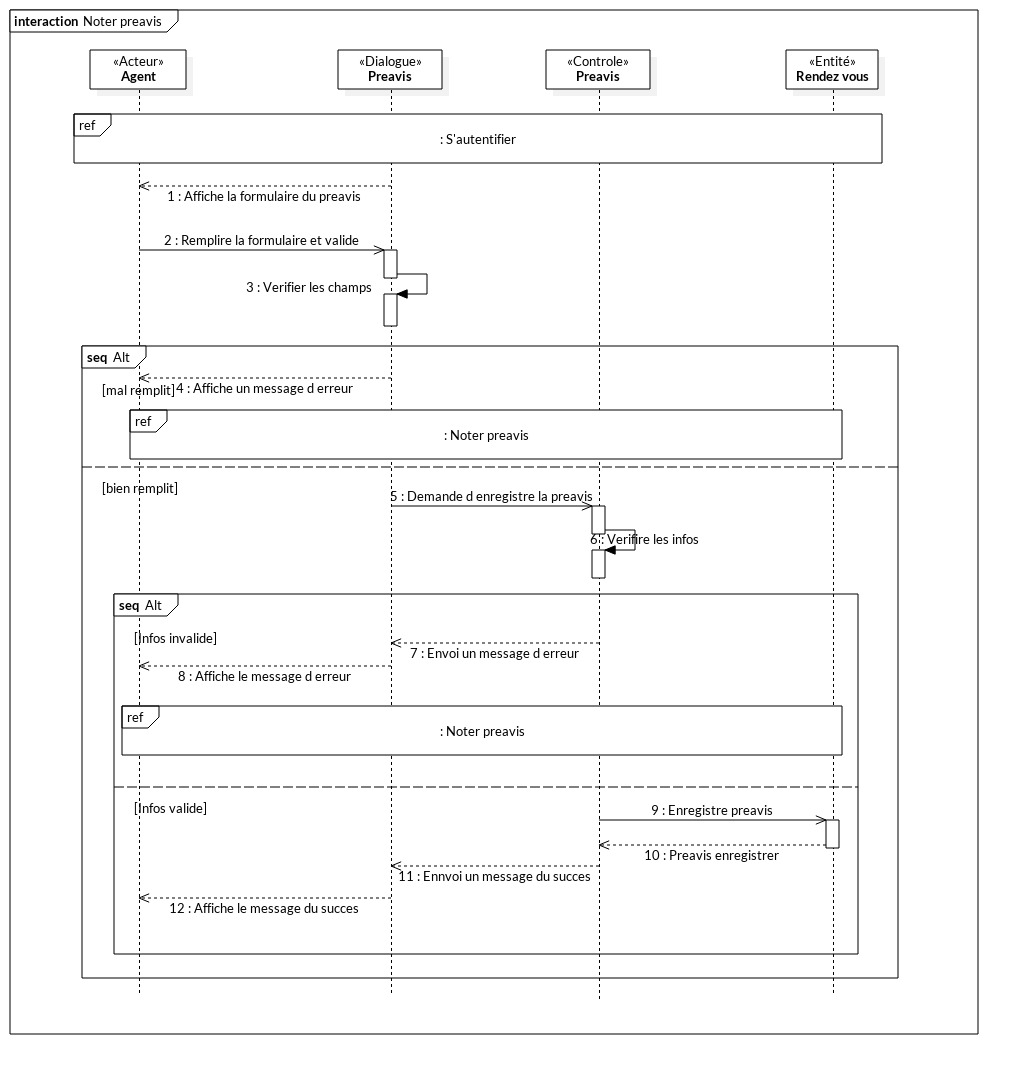
3.4-Changer rendez-vous

3.5-Singaler rendez-vous :

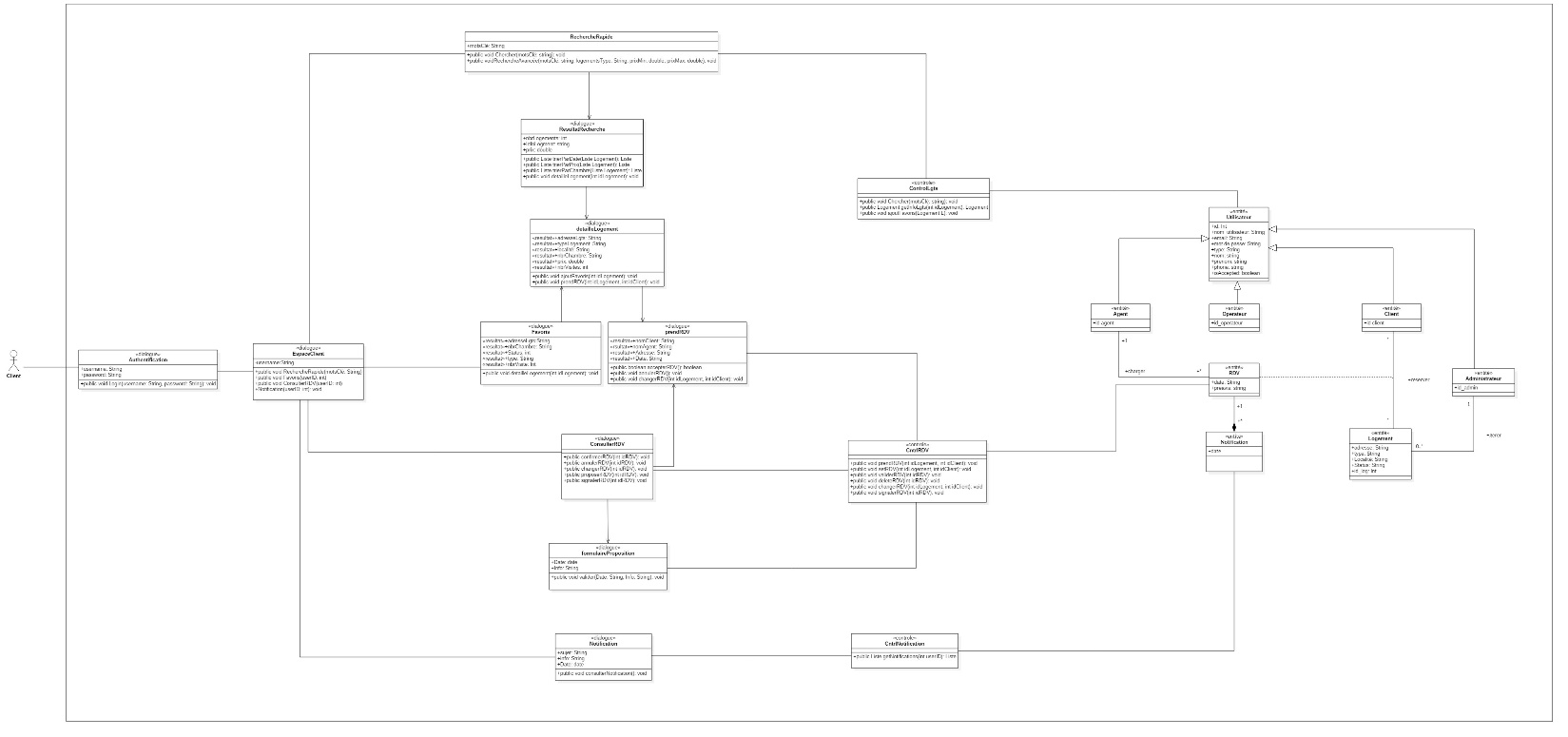
3.6- Consulter rendez-vous :

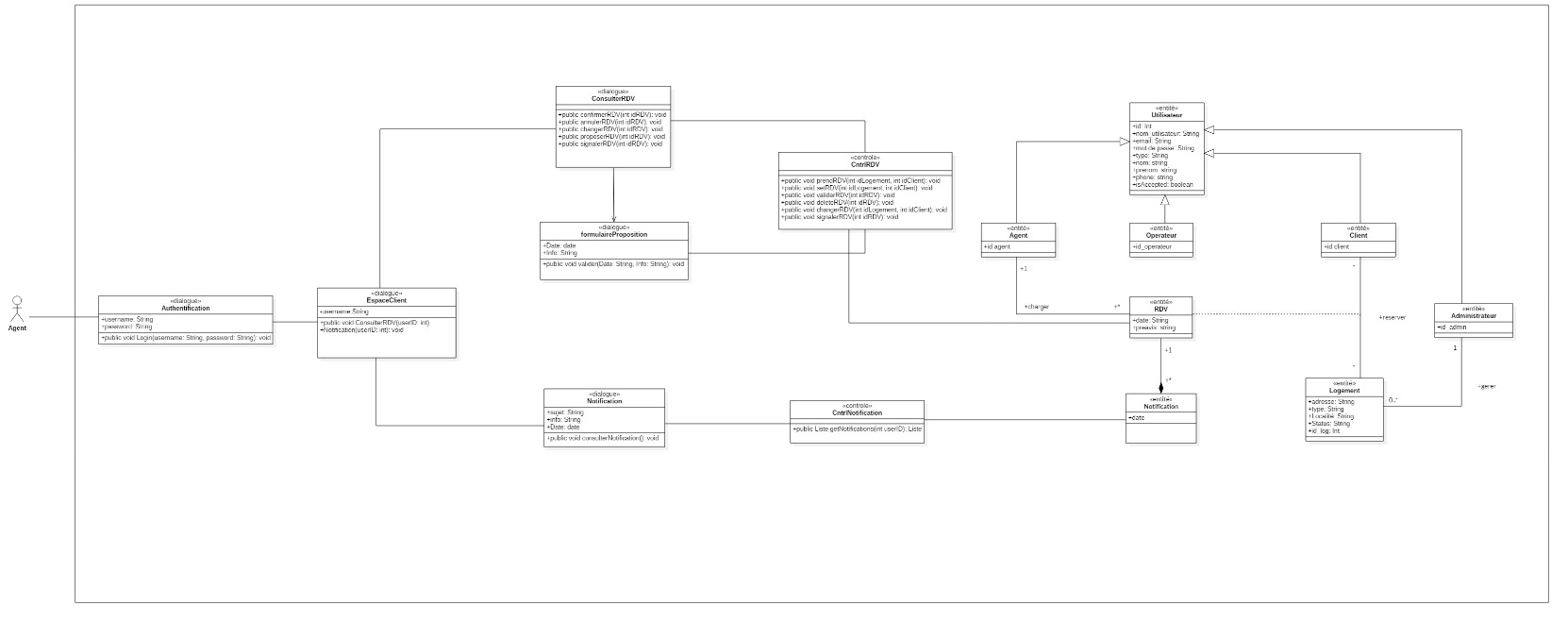
3.8- Valider l’inscription :

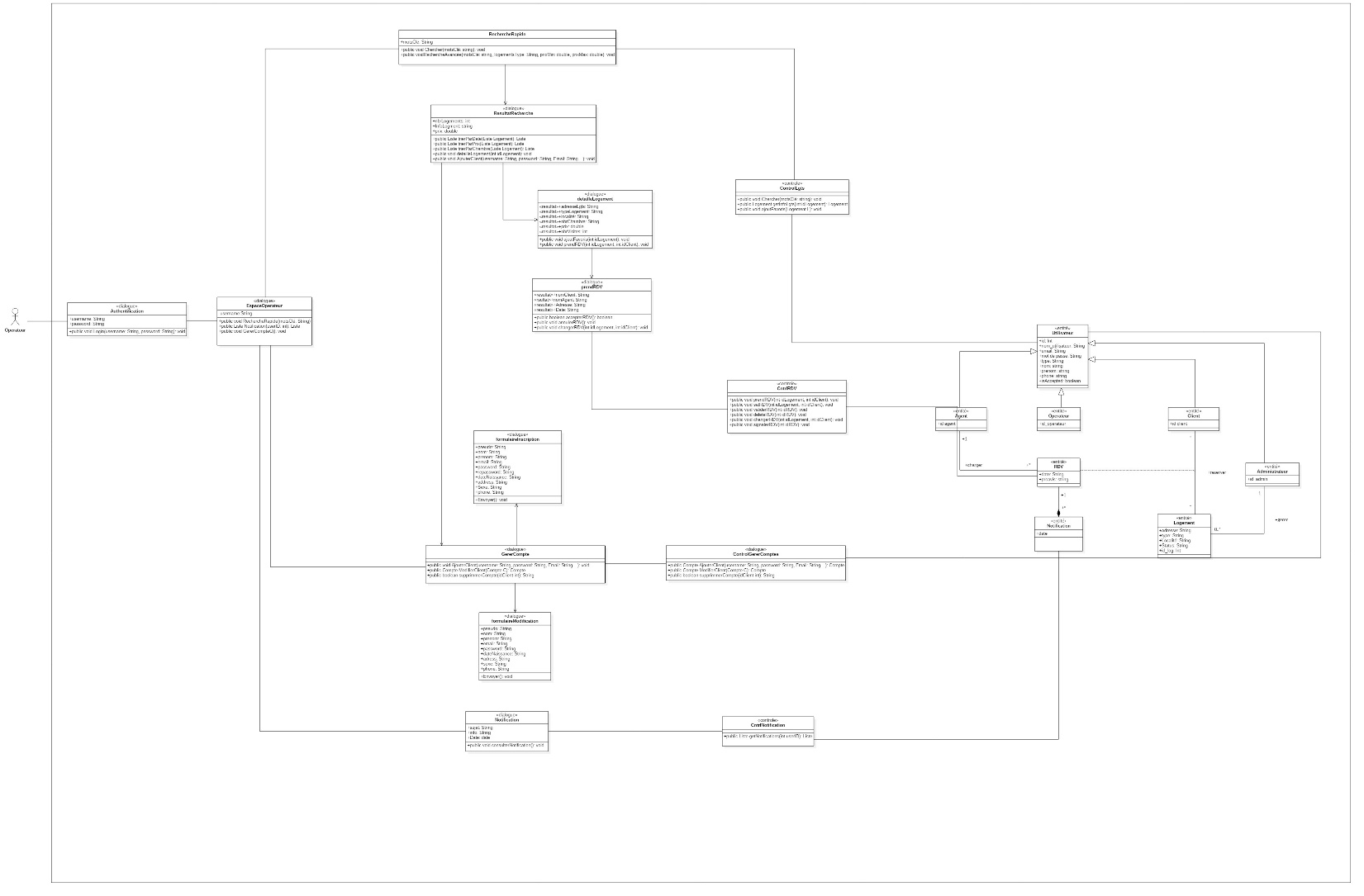


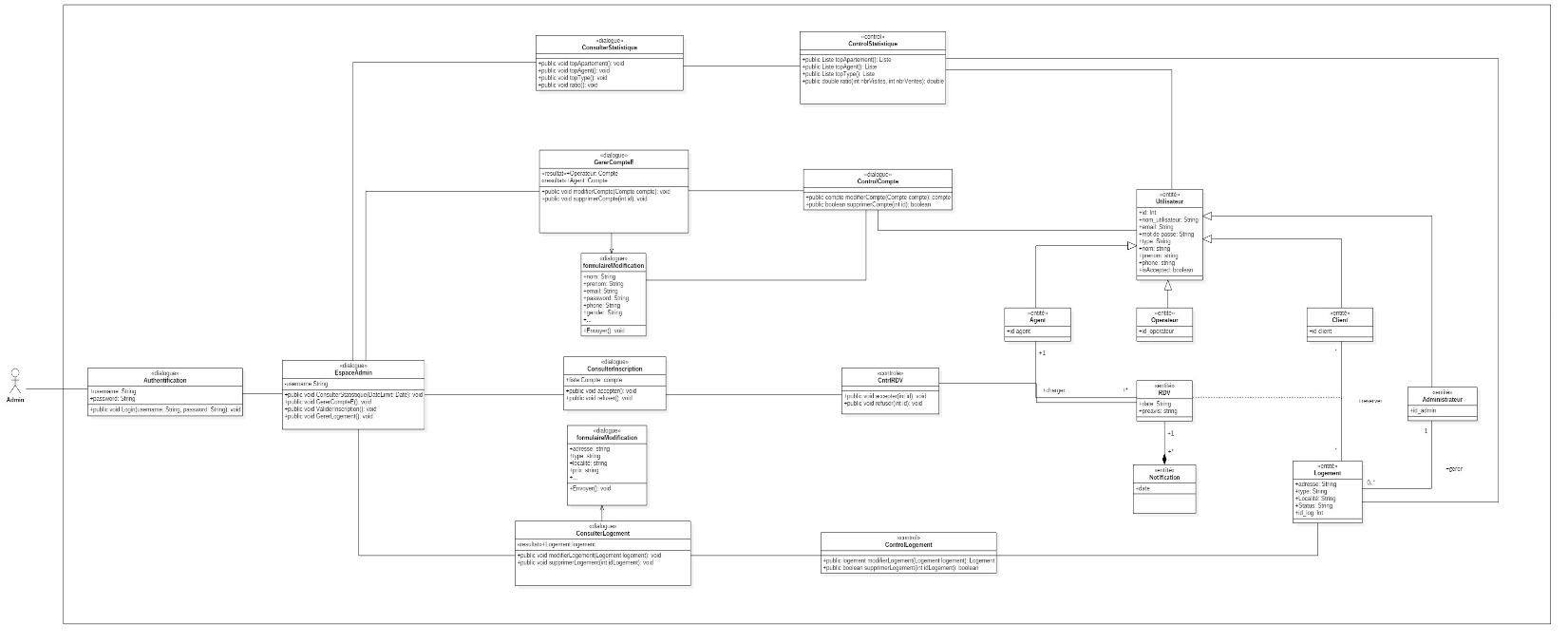
 3.9- Noter préavis :

2-Les Diagrammes des classes conception :

2-1. Client :

2.2- Agent :

2.3- Operateur :

2.4-Administrateur :

**Chapitre 3** : Réalisation

# 1-Introduction

A Ce stade de processus, il reste que la programmation et l’implémentation du site.

Dans ce chapitre nous allons décrire les différents logiciels et outils, ainsi que les langages de programmation utilisés dans la création et la mise en œuvre des différentes parties de notre application. Ainsi nous décrivons l'environnement de développement du système et de la base de données.

**2- Plateformes :**

Il y a deux types de plateformes sont à présenter ici :

- Plateforme matérielle, nous allons présenter les machines sur lesquelles nous avons réalisé et testé notre site.

- Plateforme logicielle représente les outils, logiciels et les langages de programmation orientés web utilisés.

**2.1- Plateforme matérielle :**

**- Machine Serveur :**

|  |  |
| --- | --- |
| PC | DESKTOP-Toshiba |
| Type du système | Ubuntu 16.04 |
| Processeur | Intel Corê i3-3300U |
| Mémoire installe(RAM) | 4GB |
| Disque dur | 500GB |
| Carte Graphique | Intel IV bridge Mobile |
| Carte réseaux | Athéros Ar5006X |

**- Machine Client :**

|  |  |
| --- | --- |
| PC | DESKTOP-3GVVBHB |
| Type du système | Windows 10 Pro 64-bit |
| Processeur | Intel Core i5-6200U |
| Mémoire installe(RAM) | 8GB |
| Disque dur | 500GB |
| Carte Graphique | AMD Radeon M330 |
| Carte réseaux | Realtek RTL8723BE |

# 

**2.2-Plateforme logicielle :**

**2.2.1-Environnement de développement :**

Pour développer un système qui répond aux besoins, il faut utiliser des systèmes et des logiciels complexes. On présente ici les outils de développement que nous avions utilisé.

1. Système d’exploitation :

C’est l’environnement de base qui a constitué le support de notre travail, Ubuntu est une distribution [**GNU/Linux**](http://fr.wikipedia.org/wiki/Linux), modulable et adapté à l’utilisateur.

## **Langages de programmation :**

Dans le web, un seul langage de programmation ne suffit pas pour faire fonctionner un site ou une application web de qualité, on a besoin donc d’utiliser plusieurs langages chacun son rôle.

### **HTML 5**

### HTML_100 HTML est un langage de balisage permettant d’écrire de l’hypertexte, d’où son nom. HTML permet également de structurer sémantiquement et de mettre en forme le contenu des pages, d’inclure des ressources multimédias dont des images, des formulaires de saisie, et des programmes informatiques. C’est le langage responsable sur la structure de la page web.

### **CSS 3**

Les feuilles de style en cascade, généralement appelées CSS de l'anglais « *Cascading Style Sheets* », forment un langage informatique qui décrit la présentation des documents HTML. Introduit en HTML 4, CSS est le langage responsable sur le style du site (tous ce qui concerne les couleurs et le design).

### **badges_JavaScript_Basics_Stage1_100JS :**

JavaScript est un langage de programmation de scripts principalement employé dans les pages web interactives mais aussi pour les serveurs. Son rôle est de rendre le site ou l’application plus fluide et plus facile à utiliser.

### **AJAX :**

L'architecture informatique Ajax permet de construire des applications Web et des sites web dynamiques interactifs sur le poste client en se servant de différentes technologies ajoutées aux navigateurs web. Ajax est très essentiel surtout pour les formulaires dynamiques.

### C:\Users\Cyanide\Desktop\images (1).pngXML :

### Langage extensible de balisage, est un langage de description des données. Il permet de structurer et échanger des données de type texte. Il est dit extensible car il permet à l'utilisateur de personnaliser ses balises.

### **C:\Users\Cyanide\Desktop\23086798.pngJava EE :**

Le terme « Java EE » signifie Java Enterprise Edition, et était anciennement raccourci en « J2EE » est une [spécification](https://fr.wikipedia.org/wiki/Sp%C3%A9cification_(norme_technique)) pour la [plate-forme Java](https://fr.wikipedia.org/wiki/Plate-forme_Java) d'[Oracle](https://fr.wikipedia.org/wiki/Oracle_(entreprise)), destinée aux applications d'entreprise . Il fait quant à lui référence à une extension de la plate-forme standard. L'objectif majeur de Java EE est de faciliter le développement d'applications web robustes et distribuées, déployées et exécutées sur un serveur d'applications.



### **Spring :**

Est un [Framework](https://fr.wikipedia.org/wiki/Framework) [libre](https://fr.wikipedia.org/wiki/Logiciel_libre) pour construire et définir l'infrastructure d'une application [java](https://fr.wikipedia.org/wiki/Java_(technique))[4](https://fr.wikipedia.org/wiki/Spring_(framework)#cite_note-4), dont il facilite le [développement](https://fr.wikipedia.org/wiki/D%C3%A9veloppement_de_logiciel) et les tests. Il est effectivement un conteneur dit "léger", c’est-à-dire une infrastructure similaire à un [serveur d'applications](https://fr.wikipedia.org/wiki/Serveur_d%27applications) [J2EE](https://fr.wikipedia.org/wiki/Java_EE). Il prend donc en charge la création d’objets et la mise en relation d’objets par l’intermédiaire d’un fichier de configuration qui décrit les objets à fabriquer et les relations de dépendances entre ces objets. Le gros avantage par rapport aux serveurs d’application est qu’avec Spring, les classes n’ont pas besoin d’implémenter une quelconque interface pour être prises en charge par le Framework (au contraire des [serveur d'applications](https://fr.wikipedia.org/wiki/Serveur_d%27applications) [J2EE](https://fr.wikipedia.org/wiki/Java_EE) et des [EJBs](https://fr.wikipedia.org/wiki/Enterprise_JavaBeans)). C’est en ce sens que Spring est qualifié de conteneur « léger ».

### **badge_php_laravelbasics_stage03Spring Web MVC :**

Spring Web MVC Framework fournit une architecture MVC (Model-View-Controller) et des composants prêts à l'emploi qui peuvent être utilisés pour développer des applications Web flexibles et faiblement couplées. Le modèle MVC permet de séparer les différents aspects de l'application (logique d'entrée, logique métier et logique UI) tout en assurant un couplage lâche entre ces éléments.

- Un modèle (Model) contient les données à afficher.

- Une vue (View) contient la présentation de l'interface graphique.

- Un contrôleur (Controller) contient la logique concernant les actions effectuées par l'utilisateur.

### **C:\Users\Cyanide\AppData\Local\Microsoft\Windows\INetCache\Content.Word\download.pngSpring Boot :**

Est un micro Framework qui a notamment pour but de faciliter la configuration d’un projet Spring et de réduire le temps alloué au démarrage d’un projet.

### **C:\Users\Cyanide\AppData\Local\Microsoft\Windows\INetCache\Content.Word\download.pngJava Server Pages :**

 JSP est une [technique](https://fr.wikipedia.org/wiki/Technique) basée sur [Java](https://fr.wikipedia.org/wiki/Java_(langage)) qui permet aux [développeurs](https://fr.wikipedia.org/wiki/D%C3%A9veloppeurs) de créer dynamiquement du code [HTML](https://fr.wikipedia.org/wiki/Hypertext_markup_language), [XML](https://fr.wikipedia.org/wiki/Extensible_markup_language) ou tout autre type de [page web](https://fr.wikipedia.org/wiki/Page_web). Cette technique permet au code Java et à certaines actions prédéfinies d’être ajoutés dans un contenu statique.



### **Java Server Pages Standard Tag Library :**

### JSTL est un composant de la plate-forme [JEE](https://fr.wikipedia.org/wiki/JEE) de développement. Elle étend la spécification [JSP](https://fr.wikipedia.org/wiki/JavaServer_Pages) en ajoutant une bibliothèque de balises pour les tâches courantes, comme le travail sur des fichiers [XML](https://fr.wikipedia.org/wiki/Extensible_Markup_Language), l'exécution conditionnelle,

### **Expression langage :**

Une EL permet d'accéder simplement aux beans des différents scopes de l'application web (page, requête, session et application). Utilisé conjointement avec des librairies de tags, elles permettent de se passer totalement des scriptlets.

### **badges_DD_Database_Stage2SQL** :

Afin d’exploiter notre BD, on utilise SQL qui signifie « Structured Query Language » (langage de requête structurée). C’est un langage qui permet de rechercher, d'ajouter, de modifier ou de supprimer des données dans les bases de données relationnelles.



### **Hibernate** :

Est un [Framework](https://fr.wikipedia.org/wiki/Framework) [open source](https://fr.wikipedia.org/wiki/Open_Source) gérant la [persistance](https://fr.wikipedia.org/wiki/Persistance_(informatique)) des [objets](https://fr.wikipedia.org/wiki/Programmation_orient%C3%A9e_objet) en [base de données relationnelle](https://fr.wikipedia.org/wiki/Base_de_donn%C3%A9es_relationnelle). Il est adaptable en termes d'architecture, il peut donc être utilisé aussi bien dans un [développement](https://fr.wikipedia.org/wiki/D%C3%A9veloppeur) client lourd, que dans un environnement web léger de type [Apache Tomcat](https://fr.wikipedia.org/wiki/Apache_Tomcat) ou dans un environnement [Java EE](https://fr.wikipedia.org/wiki/Java_EE) complet.

### **C:\Users\Cyanide\AppData\Local\Microsoft\Windows\INetCache\Content.Word\images.jpg La Java Persistance :**

JPA est une [interface de programmation](https://fr.wikipedia.org/wiki/Interface_de_programmation) Java permettant aux développeurs d'organiser des [données relationnelles](https://fr.wikipedia.org/wiki/Mod%C3%A8le_relationnel) dans des applications utilisant la plateforme [Java](https://fr.wikipedia.org/wiki/Java_(technique)).

## **Outils de développement :**

1. **MySQL :**

Est un système de gestion de base de données relationnelle (SGBDR). Il est distribué sous une double licence GPL. Il fait partie des logiciels de gestion de base de données les plus utilisés au monde, autant par le grand public (applications web principalement) que par les professionnels.

1. **Interface d’SGBD > PhpMyAdmin :**

PhpMyAdmin est une application web qui permet de gérer un serveur de bases de données MySQL. Dans un environnement multi-utilisateur, cette interface écrite en PHP permet également de donner à un utilisateur un accès à ses propres bases de données.

1. **Serveur Web > Apache TomCat :**

Est un [conteneur web](https://fr.wikipedia.org/wiki/Conteneur_de_servlets) [libre](https://fr.wikipedia.org/wiki/Logiciel_libre) de [servlets](https://fr.wikipedia.org/wiki/Servlet) et [JSP](https://fr.wikipedia.org/wiki/JavaServer_Pages) [Java EE](https://fr.wikipedia.org/wiki/Java_EE). Issu du projet [Jakarta](https://fr.wikipedia.org/wiki/Apache_Jakarta), c'est un des nombreux projets de l’[Apache Software Fondation](https://fr.wikipedia.org/wiki/Apache_Software_Foundation). Il implémente les spécifications des [servlets](https://fr.wikipedia.org/wiki/Servlet) et des JSP du [Java Community Process](https://fr.wikipedia.org/wiki/Java_Community_Process), est paramétrable par des fichiers [XML](https://fr.wikipedia.org/wiki/Extensible_Markup_Language) et des propriétés, et inclut des outils pour la configuration et la gestion. Il comporte également un [serveur HTTP](https://fr.wikipedia.org/wiki/Serveur_HTTP).

1. **IDE > Spring Tool Suite :**

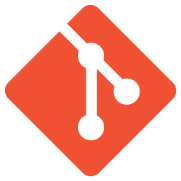
Est un environnement de développement basé sur Eclipse qui est personnalisé pour le développement d'applications Spring. Il fournit un environnement prêt à l'emploi pour implémenter, déboguer, exécuter et déployer vos applications Spring.

1. **Spring Initialiser :**

Fournit une API extensible pour générer des projets Quickstart. Il offre également un service configurable « https://start.spring.io ». Il fournit une interface utilisateur Web simple pour configurer le projet à générer et les points de terminaison que vous pouvez utiliser via Https simples.

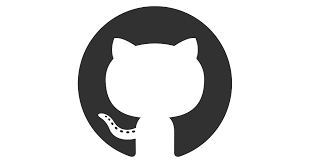
1. **Logiciel de** [**gestion de dépendances**](http://objis.com/tutoriel-maven-n4-gestion-de-dependances/) **> Apache Maven :**

  Est un outil de [gestion et d'automatisation de production des projets logiciels](https://fr.wikipedia.org/wiki/Moteur_de_production) [Java](https://fr.wikipedia.org/wiki/Java_(langage)) en général et [Java EE](https://fr.wikipedia.org/wiki/Java_EE) en particulier. Il utilise un paradigme connu sous le nom de *Project Object*

****

1. [**Logiciel de gestion de versions**](https://fr.wikipedia.org/wiki/Logiciel_de_gestion_de_versions) **> Git :**

Est un [logiciel de gestion de versions](https://fr.wikipedia.org/wiki/Logiciel_de_gestion_de_versions) [décentralisé](https://fr.wikipedia.org/wiki/Gestion_de_versions#Gestion_de_versions_d.C3.A9centralis.C3.A9e). Un système qui permet le développement facile des projets informatiques. Avec ce système, plusieurs développeurs peuvent travailler sur un projet sans qu’il y’aurait des ambigüités, ni de problèmes de versions de code. Le code sera open source [[1]](#endnote-1)afin que les développeurs du monde entier puissent le consulter, commenter et l’améliorer en proposant une nouvelle branche de développement.



1. **GitubHub :**

Est un service web d'[hébergement](https://fr.wikipedia.org/wiki/H%C3%A9bergeur_web) et de gestion de développement de logiciels, utilisant le [logiciel de gestion de versions](https://fr.wikipedia.org/wiki/Logiciel_de_gestion_de_versions) [Git](https://fr.wikipedia.org/wiki/Git)

****

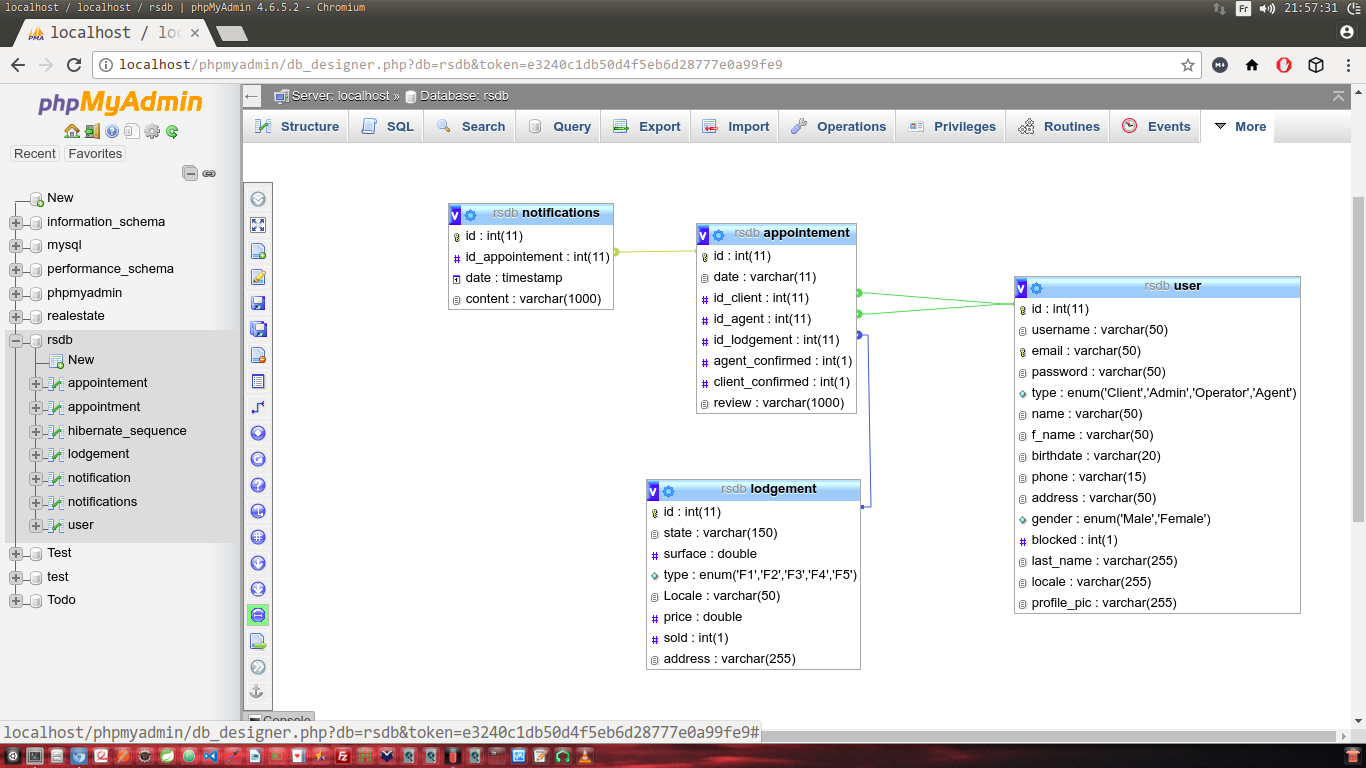
1. **Chrome:**

Est un navigateur web propriétaire développé par Google basé sur le projet libre Chromium fonctionnant sous Windows, Mac, Linux, Android et iOS.

# Présentation de notre application :

3.1- Réalisation de la base de données :

La base de données que nous allons utiliser contient (4) tables pour conserver les différentes informations nécessaires pour l’invocation des fonctionnalités du site.



-la structure de notre base de données-

3.2- Example Interface:

# Conclusion

Dans ce dernier chapitre, nous avons présenté la structure de notre système, les outils utilisés et la façon dont on a intégré les nouvelles notions et les bonnes pratiques pour produire un résultat propre réutilisable et maintenable. Cela a été aussi l’occasion de montrer les différentes parties de l’application développée par quelques captures d’écran.

1. [↑](#endnote-ref-1)